DJANGO WEB DEVOLOPMENT PROJECT

Documentation

- Aman Bele

Note:

This documentation is intended to be followed with Python **v3.10** and above **ONLY**.

There might be some issues with old versions below v3.9

Also, this documentation is meant to be executed with native windows CMD. Below programs are its major incompatibilities:

PyCharm v2020 and below *(Professional and Community Versions)*

Sublime Text 3 and below

I highly recommend that you should use **Windows 10** if you are still on Windows 7. More info with updated information can be found at this link. I also recommend that you should update your programming tools to their latest version for the best stability and performance.

My System Configurations and Information

Operating System: Windows 10 Home 21H2

Code Editor: Sublime Text 4 *(Build 4126)*

Terminal: Windows 10 CMD

Python: Python v3.10

My Project Configurations

Django: v4.0.4

INDEX:

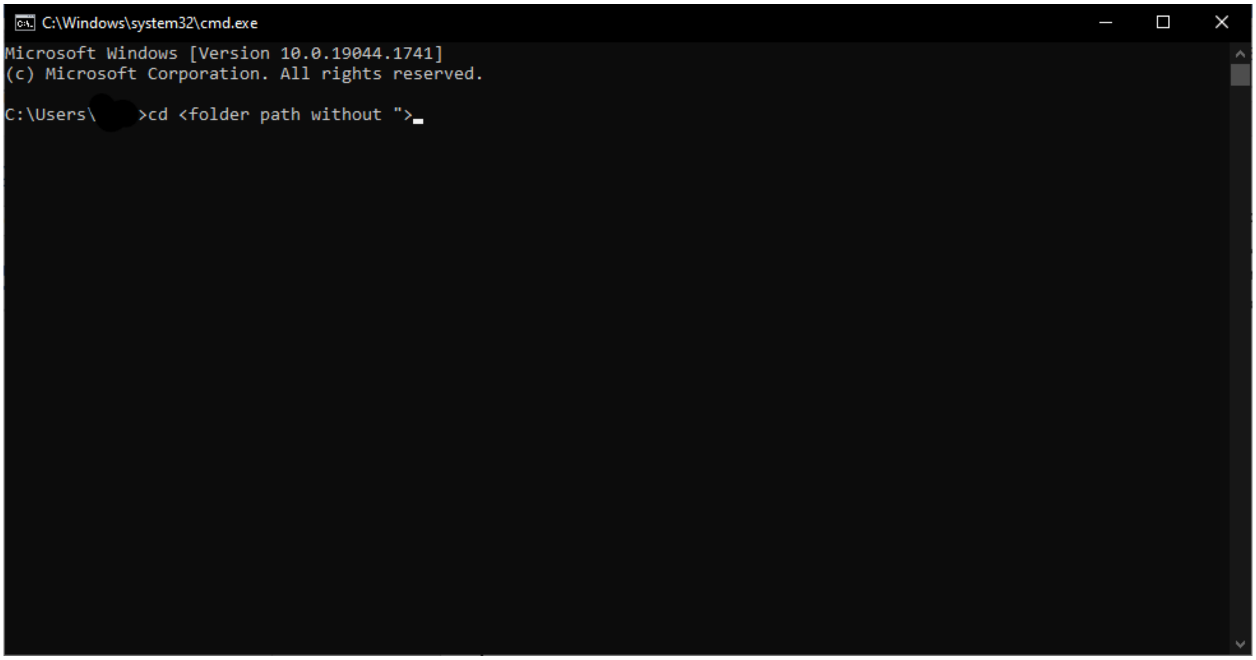
Django E-Commerce Website Project

Procedural Instructions *(GitHub Reference at the last of the document)*

Step 1 | Installation

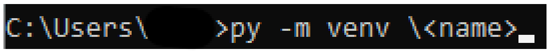
Creating a virtual environment

Creating a virtual environment is recommended as it makes it easy to install multiple python plugins, if necessary, without changing the global settings. In short, it makes a virtual environment for all your needs. First, create a new folder for your project. In this case, I'll create a folder named shop. Then open a new cmd window inside the new folder you've created by hitting Win + R and typing cmd and hitting Ok.



```
Microsoft Windows [Version 10.0.19044.1741]
(c) Microsoft Corporation. All rights reserved.

C:\Users\[redacted]>cd <folder path without ">
```

Then type the command above and replace <folder path without "> with your folder path. Then type the following command and replace <name> with your project name.



```
C:\Users\[redacted]>py -m venv \<name>
```

This will create a folder named venv in your project folder.

As of updated python versions, there's no need to activate the virtual environment but still, we can activate it by running the activate.bat script inside the venv folder.

Now you've successfully activated the virtual environment and your cmd should have (venv) <project path> as its first line.

Installing Django

Type py -m pip install Django inside the cmd window. This will install Django inside your virtual environment *(if you have one)*. Now run django-admin startproject <project name>. This will create a project folder named with your project name *(here it's shop)*. Now your directory should look something like this. "." Is root directory.

```
.
└── <project_name>
    ├── <project>
    │   ├── __init__.py
    │   ├── asgi.py
    │   ├── settings.py
    │   ├── urls.py
    │   └── wsgi.py
    └── manage.py
```

Now you've set up all the files needed to make this project a successful one! Let's go!

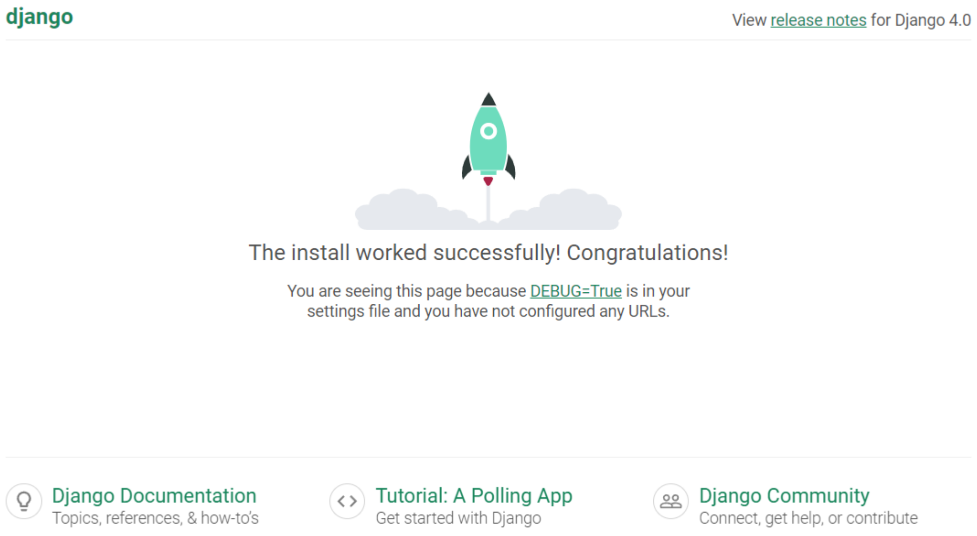
Adding the app *app*

Now open a cmd window in <project_name> folder and type py manage.py startapp app. This will create a new folder and your directory should look something like this.

```
.
└── <project_name>
    ├── app
    │   ├── migrations
    │   ├── __init__.py
    │   ├── admin.py
    │   ├── apps.py
    │   ├── models.py
    │   ├── tests.py
    │   └── views.py
    ├── <project>
    │   ├── __init__.py
    │   ├── asgi.py
    │   ├── settings.py
    │   ├── urls.py
    │   └── wsgi.py
    └── manage.py
```

Adding remaining assets and assets folder

Before we move any further, I would like you to test your project by opening a cmd at the location where **manage.py** is located and open a cmd window here (PowerShell will work as well), and type in **py manage.py runserver**. This will start a development server at your local machine and will be hosted at localhost:8000. It should display something like this.



django

View release notes for Django 4.0

The install worked successfully! Congratulations!

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

Django Documentation
Topics, references, & how-to's

Tutorial: A Polling App
Get started with Django

Django Community
Connect, get help, or contribute

Now that you've created everything to run the project, it's now time to add the assets to your project. You can get the files I've used on my GitHub page or add your files. Copy the directory structure given below.

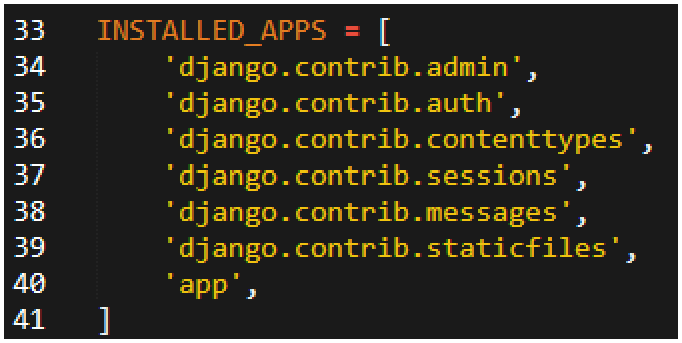
```
.
└── <project_name>
    ├── app
    │   ├── migrations
    │   ├── __init__.py
    │   ├── admin.py
    │   ├── apps.py
    │   ├── models.py
    │   ├── tests.py
    │   └── views.py
    ├── media
    │   └── productimg
    │         └── <your image files> <- This isn’t a folder or a file
    ├── <project>
    │    ├── __init__.py
    │    ├── asgi.py
    │    ├── settings.py
    │    ├── urls.py
    │    └── wsgi.py
    └── manage.py
```

Step 2 | Coding

Editing settings.py

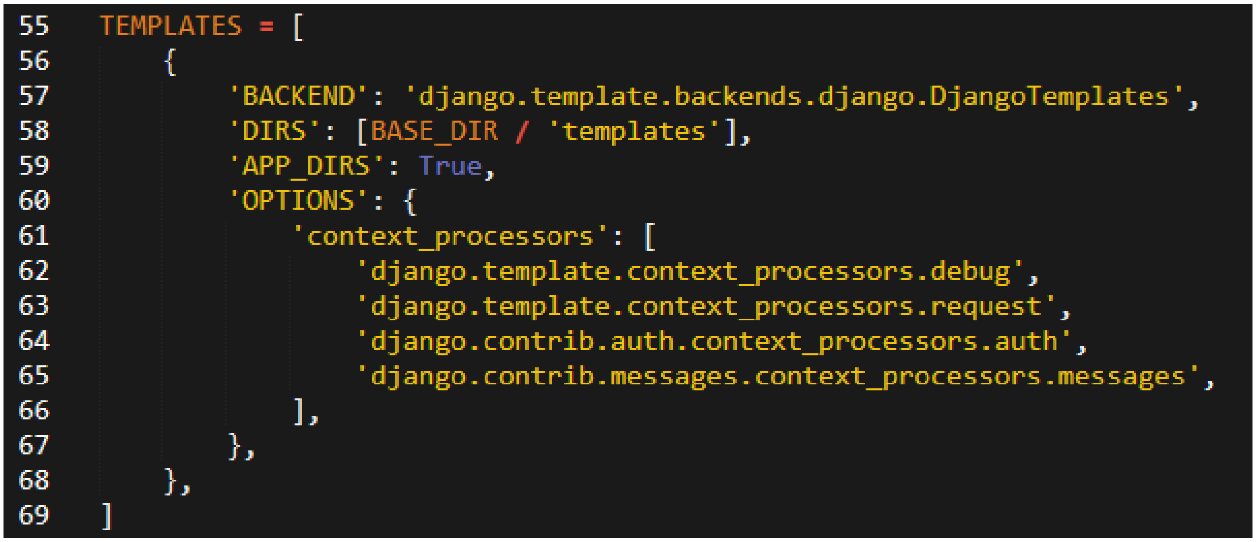
Now that you’ve set up the stuff, it’s time to edit the **settings.py** file. Refer to my GitHub for more details on it.

Add this (40) line to the **INSTALLED_APPS** block



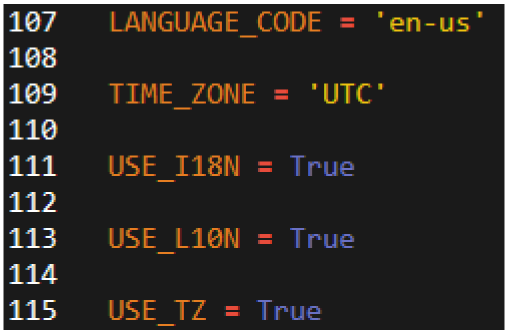
```
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'app',
41  ]
```

Add this (58) line to the **TEMPLATES** block



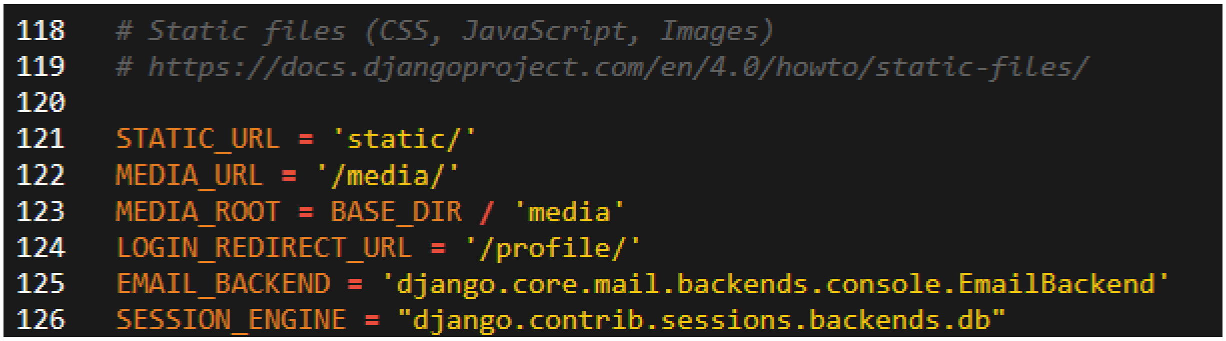
```python
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
               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

Add this (113) line to the last bit of code



```python
107  LANGUAGE_CODE = 'en-us'
108
109  TIME_ZONE = 'UTC'
110
111  USE_I18N = True
112
113  USE_L10N = True
114
115  USE_TZ = True
```

Add this before DEFAULT_AUTO_FIELD



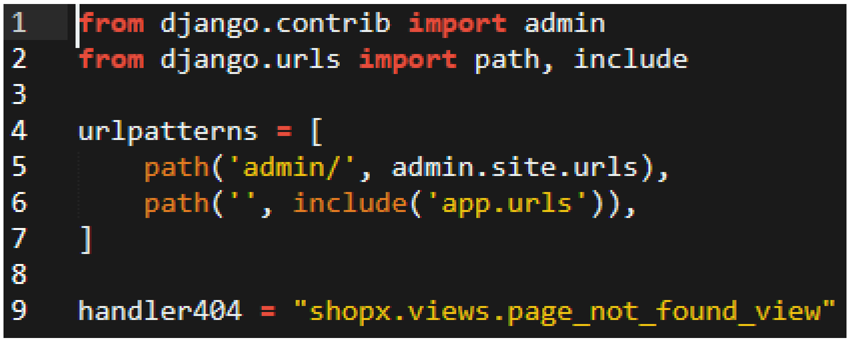
```python
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.0/howto/static-files/

STATIC_URL = 'static/'
MEDIA_URL = '/media/'
MEDIA_ROOT = BASE_DIR / 'media'
LOGIN_REDIRECT_URL = '/profile/'
EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
SESSION_ENGINE = "django.contrib.sessions.backends.db"
```

Now you've completely edited the settings.py file to your needs

Editing urls.py

Now we will edit the urls.py file

Edit the file to resemble this

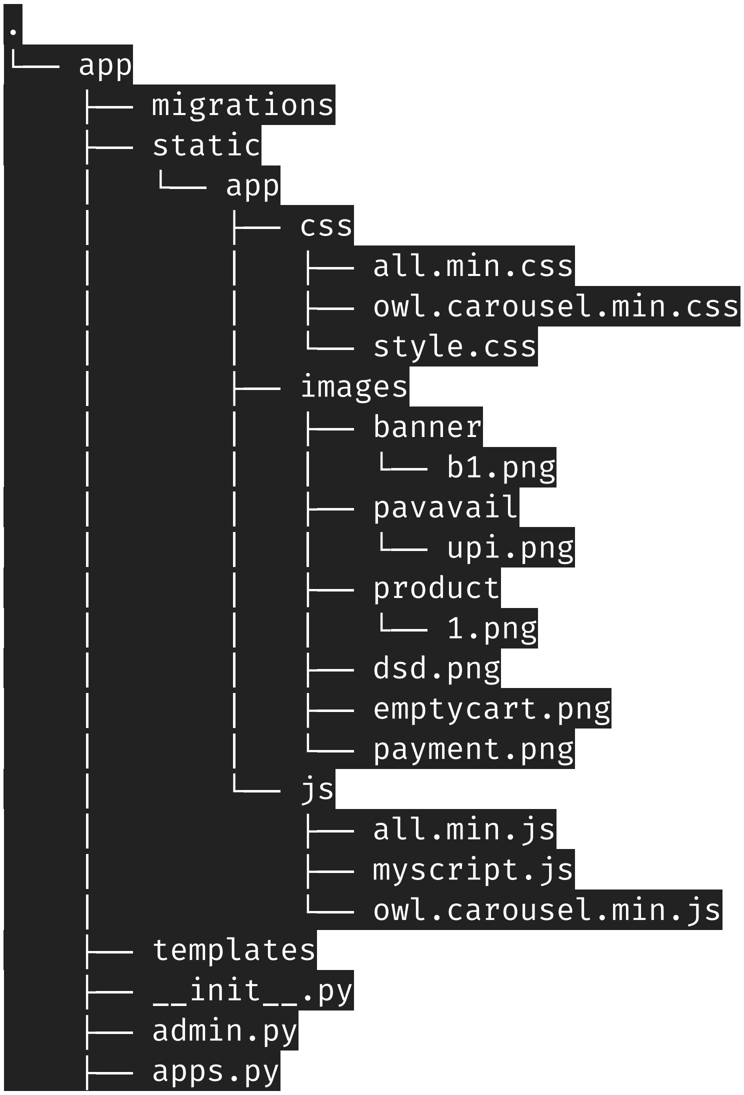


```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('app.urls')),
]

handler404 = "shopx.views.page_not_found_view"
```

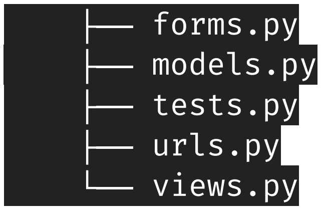
Step 3 | Working with the *apps* folder

Creating files needed

Copy the directory provided below to add the files needed. You could copy the files off my GitHub.



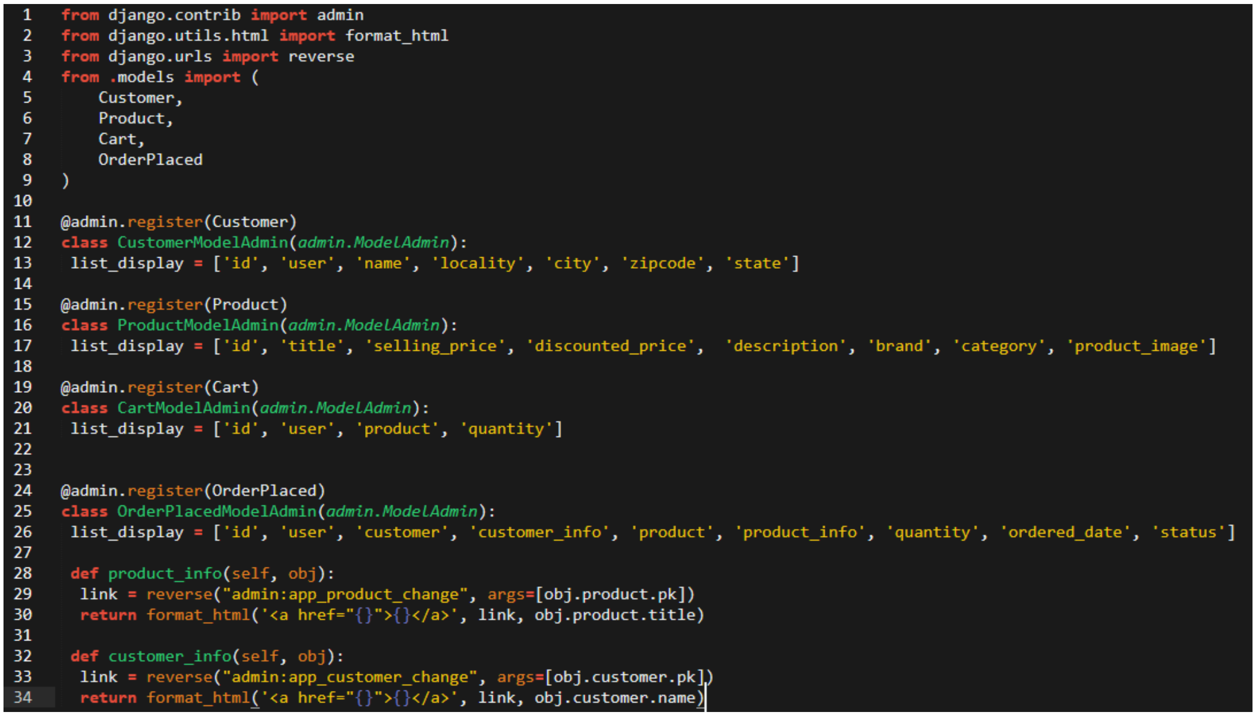
```
.
└── app
    ├── migrations
    ├── static
    │   └── app
    │       ├── css
    │       │   ├── all.min.css
    │       │   ├── owl.carousel.min.css
    │       │   └── style.css
    │       ├── images
    │       │   ├── banner
    │       │   │   └── b1.png
    │       │   ├── pavavail
    │       │   │   └── upi.png
    │       │   ├── product
    │       │   │   └── 1.png
    │       │   ├── dsd.png
    │       │   ├── emptycart.png
    │       │   └── payment.png
    │       └── js
    │           ├── all.min.js
    │           ├── myscript.js
    │           └── owl.carousel.min.js
    ├── templates
    ├── __init__.py
    ├── admin.py
    ├── apps.py
```



```
    ├── forms.py
    ├── models.py
    ├── tests.py
    ├── urls.py
    └── views.py
```

Editing admin.py

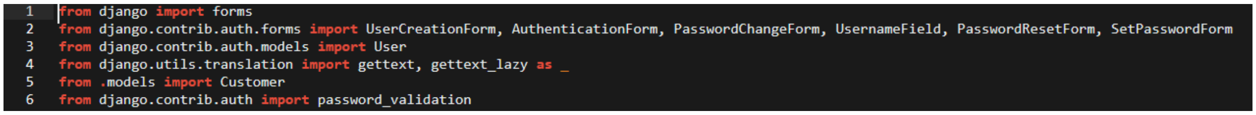
Add these lines to your **admin.py** file



```python
from django.contrib import admin
from django.utils.html import format_html
from django.urls import reverse
from .models import (
    Customer,
    Product,
    Cart,
    OrderPlaced
)

@admin.register(Customer)
class CustomerModelAdmin(admin.ModelAdmin):
 list_display = ['id', 'user', 'name', 'locality', 'city', 'zipcode', 'state']

@admin.register(Product)
class ProductModelAdmin(admin.ModelAdmin):
 list_display = ['id', 'title', 'selling_price', 'discounted_price',  'description', 'brand', 'category', 'product_image']

@admin.register(Cart)
class CartModelAdmin(admin.ModelAdmin):
 list_display = ['id', 'user', 'product', 'quantity']


@admin.register(OrderPlaced)
class OrderPlacedModelAdmin(admin.ModelAdmin):
 list_display = ['id', 'user', 'customer', 'customer_info', 'product', 'product_info', 'quantity', 'ordered_date', 'status']

 def product_info(self, obj):
  link = reverse("admin:app_product_change", args=[obj.product.pk])
  return format_html('<a href="{}">{}</a>', link, obj.product.title)

 def customer_info(self, obj):
  link = reverse("admin:app_customer_change", args=[obj.customer.pk])
  return format_html('<a href="{}">{}</a>', link, obj.customer.name)
```

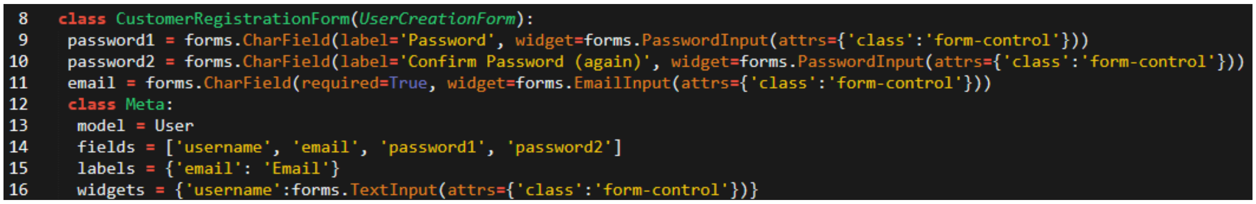
Editing forms.py

Now we will add forms to our project. We will start by importing all the necessary modules. Add these lines at the start of the document and *replace any existing ones (optional)*



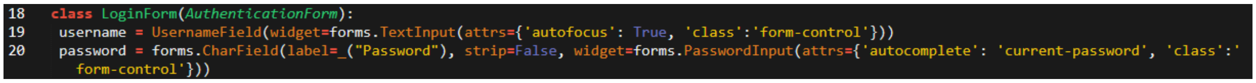
```python
from django import forms
from django.contrib.auth.forms import UserCreationForm, AuthenticationForm, PasswordChangeForm, UsernameField, PasswordResetForm, SetPasswordForm
from django.contrib.auth.models import User
from django.utils.translation import gettext, gettext_lazy as _
from .models import Customer
from django.contrib.auth import password_validation
```

Now we will start adding our classes. First off let's add **CustomerRegistrationForm.** This will be used in the customer's registration process.



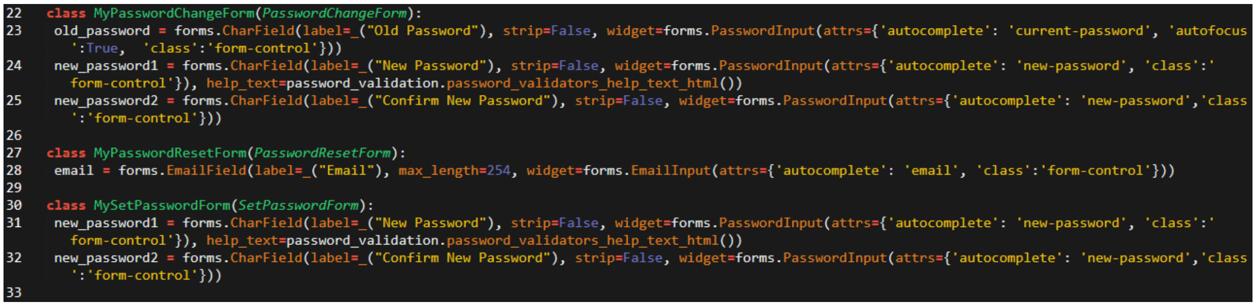
```python
class CustomerRegistrationForm(UserCreationForm):
 password1 = forms.CharField(label='Password', widget=forms.PasswordInput(attrs={'class':'form-control'}))
 password2 = forms.CharField(label='Confirm Password (again)', widget=forms.PasswordInput(attrs={'class':'form-control'}))
 email = forms.CharField(required=True, widget=forms.EmailInput(attrs={'class':'form-control'}))
 class Meta:
  model = User
  fields = ['username', 'email', 'password1', 'password2']
  labels = {'email': 'Email'}
  widgets = {'username':forms.TextInput(attrs={'class':'form-control'})}
```

Second, we will add **LoginForm.** As the name suggests, this will be used in the login process.



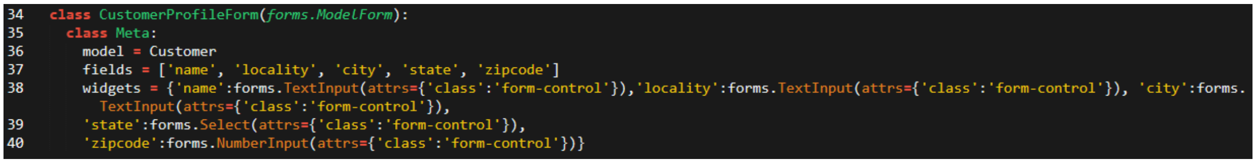
```python
class LoginForm(AuthenticationForm):
 username = UsernameField(widget=forms.TextInput(attrs={'autofocus': True, 'class':'form-control'}))
 password = forms.CharField(label=_("Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'current-password', 'class':'form-control'}))
```

Now we will add classes to manage passwords, such as **changing, resetting, and setting a new password**. Add these lines



```python
class MyPasswordChangeForm(PasswordChangeForm):
 old_password = forms.CharField(label=_("Old Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'current-password', 'autofocus':True,  'class':'form-control'}))
 new_password1 = forms.CharField(label=_("New Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'new-password', 'class':'form-control'}), help_text=password_validation.password_validators_help_text_html())
 new_password2 = forms.CharField(label=_("Confirm New Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'new-password','class':'form-control'}))

class MyPasswordResetForm(PasswordResetForm):
 email = forms.EmailField(label=_("Email"), max_length=254, widget=forms.EmailInput(attrs={'autocomplete': 'email', 'class':'form-control'}))

class MySetPasswordForm(SetPasswordForm):
 new_password1 = forms.CharField(label=_("New Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'new-password', 'class':'form-control'}), help_text=password_validation.password_validators_help_text_html())
 new_password2 = forms.CharField(label=_("Confirm New Password"), strip=False, widget=forms.PasswordInput(attrs={'autocomplete': 'new-password','class':'form-control'}))

```

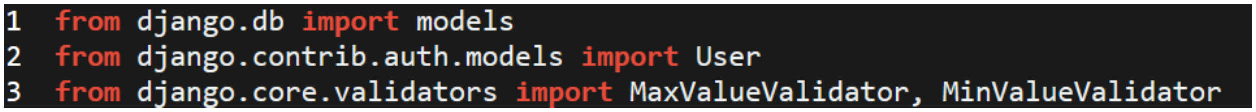
Now we will add **CustomerProfileForm**



```python
class CustomerProfileForm(forms.ModelForm):
  class Meta:
    model = Customer
    fields = ['name', 'locality', 'city', 'state', 'zipcode']
    widgets = {'name':forms.TextInput(attrs={'class':'form-control'}),'locality':forms.TextInput(attrs={'class':'form-control'}), 'city':forms.TextInput(attrs={'class':'form-control'}),
    'state':forms.Select(attrs={'class':'form-control'}),
    'zipcode':forms.NumberInput(attrs={'class':'form-control'})}
```

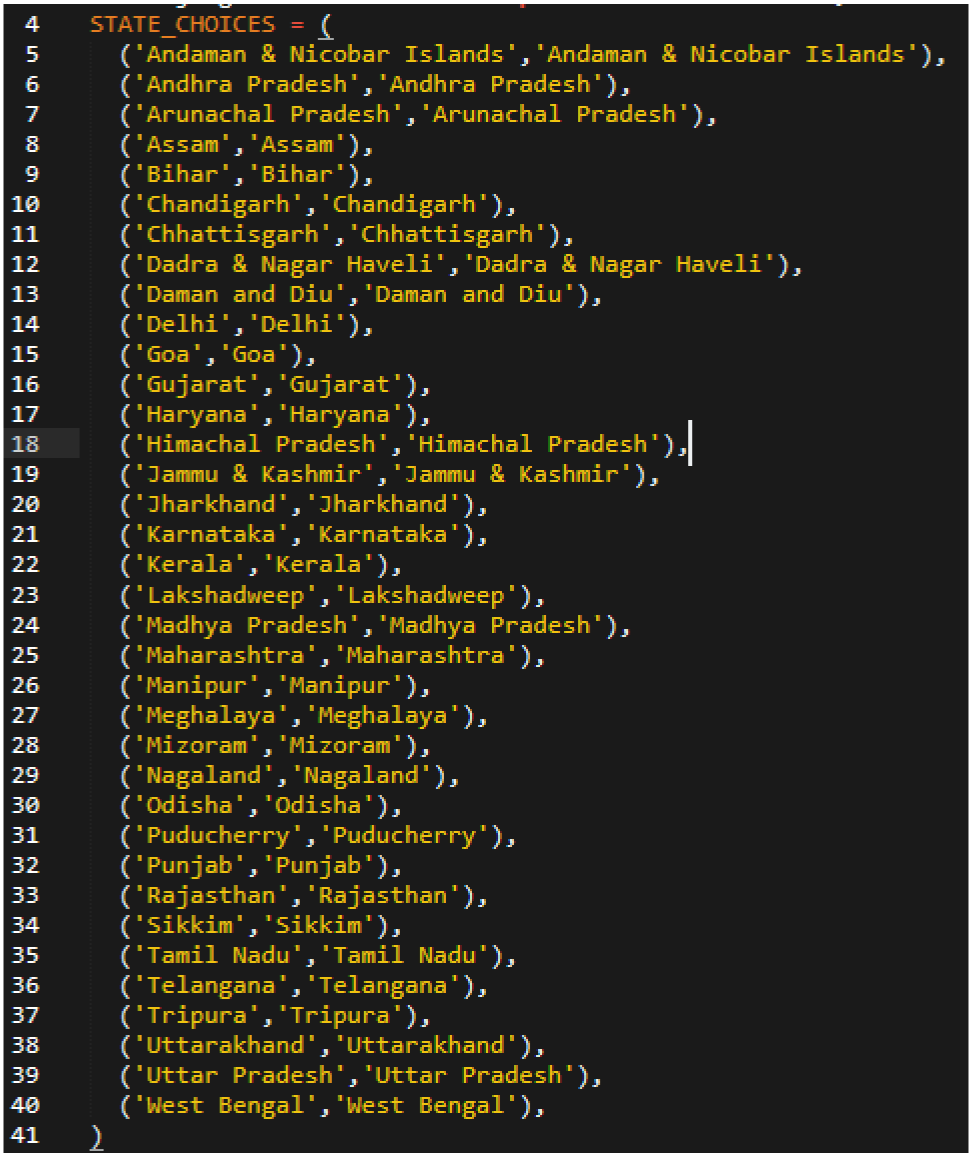
Editing models.py

Now we will start editing the **models.py** file. Start with importing the necessary modules. Add these lines to the start of the file.



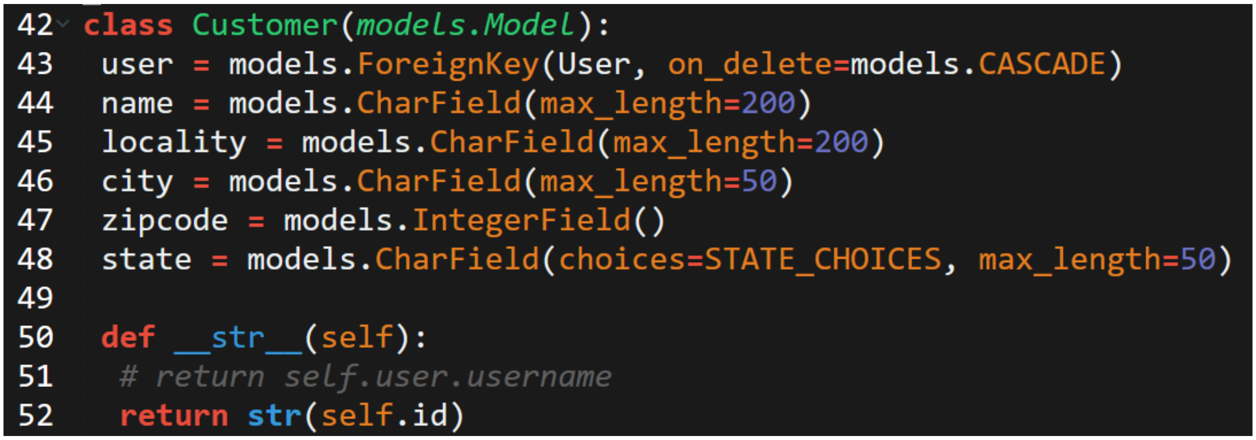
```python
from django.db import models
from django.contrib.auth.models import User
from django.core.validators import MaxValueValidator, MinValueValidator
```

Now we will add a tuple with all states from India.



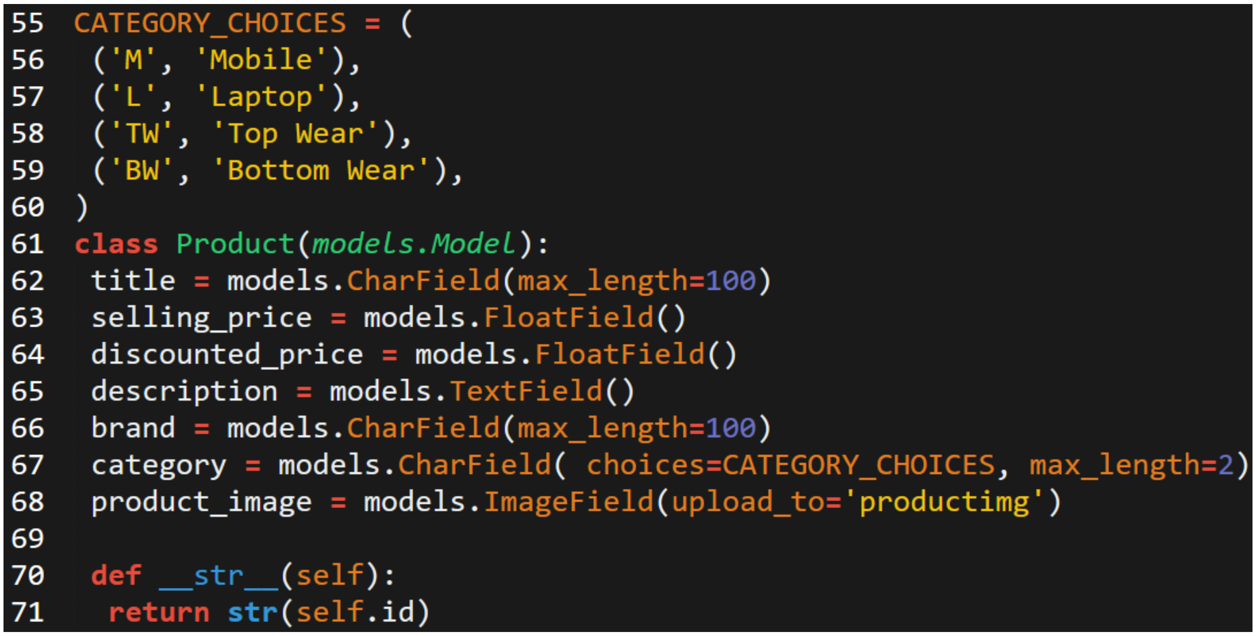
```
STATE_CHOICES = (
    ('Andaman & Nicobar Islands','Andaman & Nicobar Islands'),
    ('Andhra Pradesh','Andhra Pradesh'),
    ('Arunachal Pradesh','Arunachal Pradesh'),
    ('Assam','Assam'),
    ('Bihar','Bihar'),
    ('Chandigarh','Chandigarh'),
    ('Chhattisgarh','Chhattisgarh'),
    ('Dadra & Nagar Haveli','Dadra & Nagar Haveli'),
    ('Daman and Diu','Daman and Diu'),
    ('Delhi','Delhi'),
    ('Goa','Goa'),
    ('Gujarat','Gujarat'),
    ('Haryana','Haryana'),
    ('Himachal Pradesh','Himachal Pradesh'),
    ('Jammu & Kashmir','Jammu & Kashmir'),
    ('Jharkhand','Jharkhand'),
    ('Karnataka','Karnataka'),
    ('Kerala','Kerala'),
    ('Lakshadweep','Lakshadweep'),
    ('Madhya Pradesh','Madhya Pradesh'),
    ('Maharashtra','Maharashtra'),
    ('Manipur','Manipur'),
    ('Meghalaya','Meghalaya'),
    ('Mizoram','Mizoram'),
    ('Nagaland','Nagaland'),
    ('Odisha','Odisha'),
    ('Puducherry','Puducherry'),
    ('Punjab','Punjab'),
    ('Rajasthan','Rajasthan'),
    ('Sikkim','Sikkim'),
    ('Tamil Nadu','Tamil Nadu'),
    ('Telangana','Telangana'),
    ('Tripura','Tripura'),
    ('Uttarakhand','Uttarakhand'),
    ('Uttar Pradesh','Uttar Pradesh'),
    ('West Bengal','West Bengal'),
)
```

Now we will add the **Customer** class. This will be used to define the database entries.



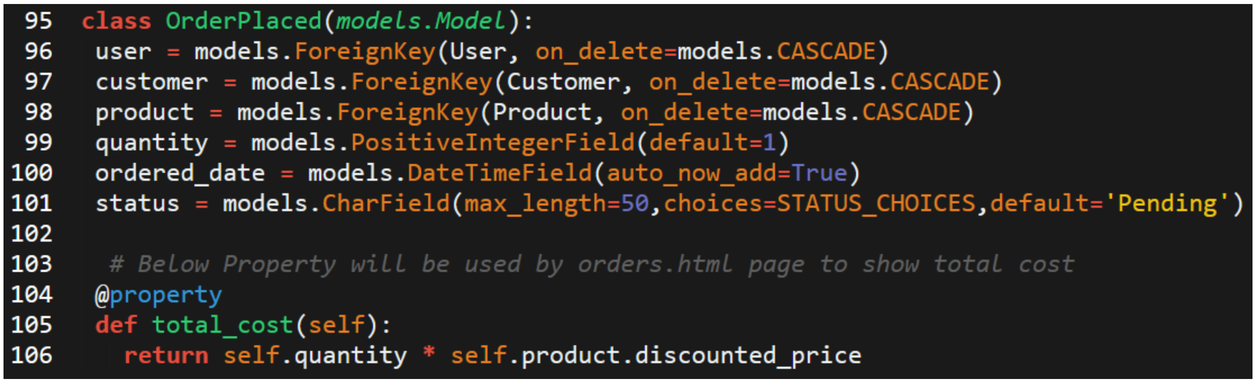
```python
class Customer(models.Model):
 user = models.ForeignKey(User, on_delete=models.CASCADE)
 name = models.CharField(max_length=200)
 locality = models.CharField(max_length=200)
 city = models.CharField(max_length=50)
 zipcode = models.IntegerField()
 state = models.CharField(choices=STATE_CHOICES, max_length=50)

 def __str__(self):
  # return self.user.username
  return str(self.id)
```

Now we will add the **CATEGORY_CHOICES** tuple and **Product** class.



```python
CATEGORY_CHOICES = (
 ('M', 'Mobile'),
 ('L', 'Laptop'),
 ('TW', 'Top Wear'),
 ('BW', 'Bottom Wear'),
)
class Product(models.Model):
 title = models.CharField(max_length=100)
 selling_price = models.FloatField()
 discounted_price = models.FloatField()
 description = models.TextField()
 brand = models.CharField(max_length=100)
 category = models.CharField( choices=CATEGORY_CHOICES, max_length=2)
 product_image = models.ImageField(upload_to='productimg')

 def __str__(self):
  return str(self.id)
```

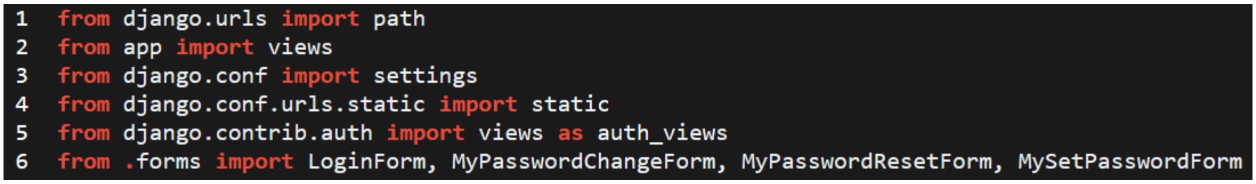
Now we will add **OrderPlaced** class and the **property**. Note that the image shows a comment stating the use of **orders.html**. We will create that file later.



```python
class OrderPlaced(models.Model):
 user = models.ForeignKey(User, on_delete=models.CASCADE)
 customer = models.ForeignKey(Customer, on_delete=models.CASCADE)
 product = models.ForeignKey(Product, on_delete=models.CASCADE)
 quantity = models.PositiveIntegerField(default=1)
 ordered_date = models.DateTimeField(auto_now_add=True)
 status = models.CharField(max_length=50,choices=STATUS_CHOICES,default='Pending')

  # Below Property will be used by orders.html page to show total cost
 @property
 def total_cost(self):
   return self.quantity * self.product.discounted_price
```

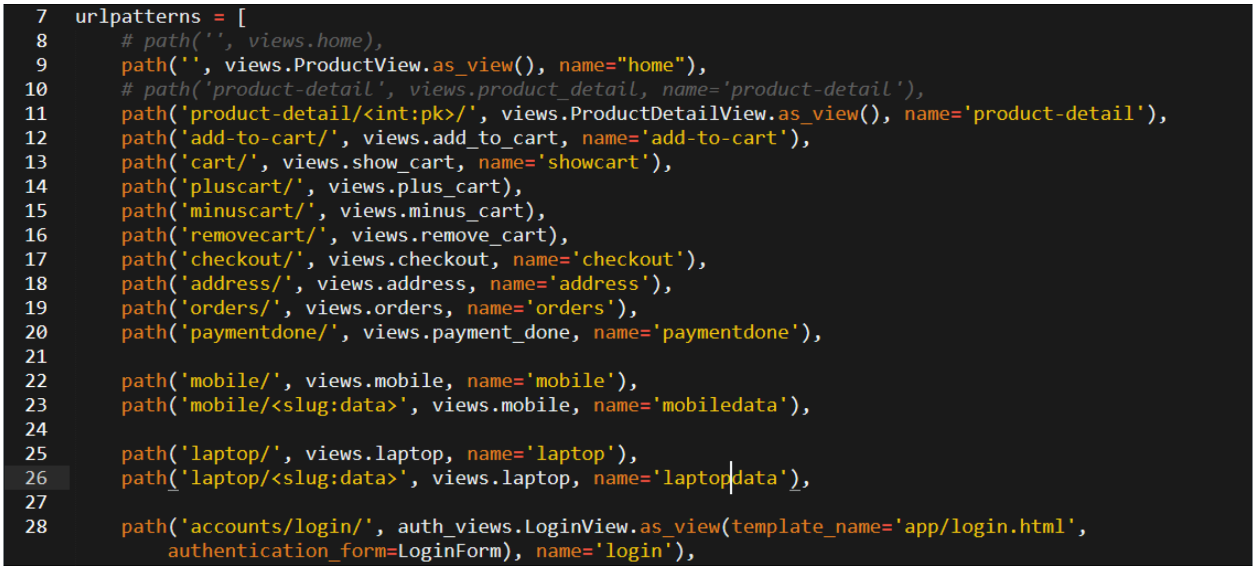
Editing urls.py

Now that we have created almost all of the backend, now we need to link the … well, links to show the website's interface *(created later on)* on specified web addresses. First off, we will import the necessary stuff needed to write the urls.py file in the first place :D. Add these lines at the start.

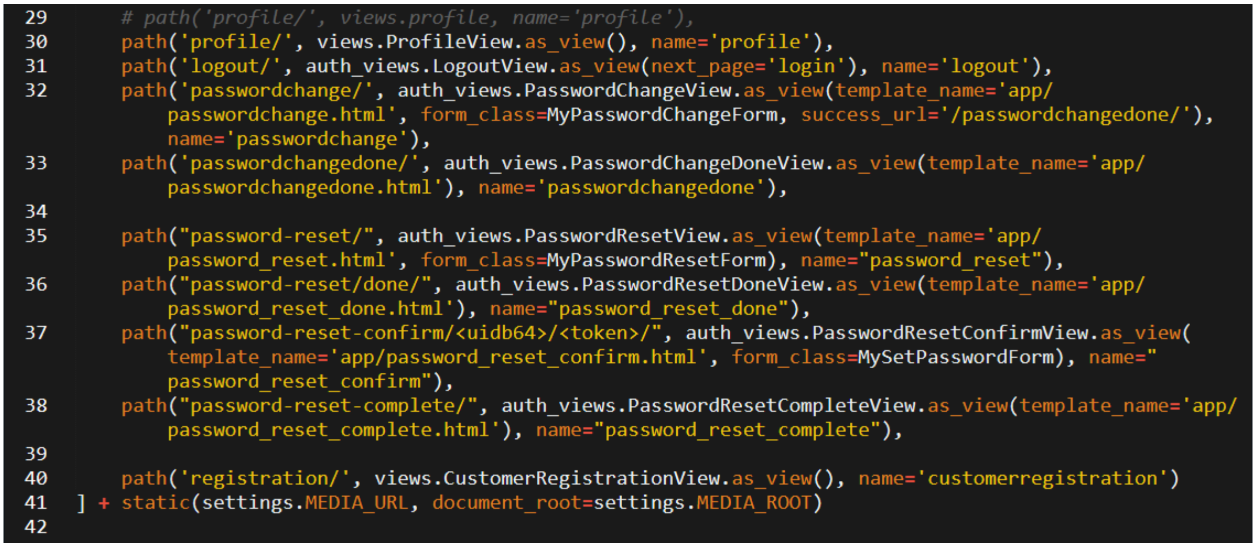


```python
from django.urls import path
from app import views
from django.conf import settings
from django.conf.urls.static import static
from django.contrib.auth import views as auth_views
from .forms import LoginForm, MyPasswordChangeForm, MyPasswordResetForm, MySetPasswordForm
```

Then we need to specify all the links with redirects to a special list named urlpatterns. Add them like below mentioned



```
urlpatterns = [
    # path('', views.home),
    path('', views.ProductView.as_view(), name="home"),
    # path('product-detail', views.product_detail, name='product-detail'),
    path('product-detail/<int:pk>/', views.ProductDetailView.as_view(), name='product-detail'),
    path('add-to-cart/', views.add_to_cart, name='add-to-cart'),
    path('cart/', views.show_cart, name='showcart'),
    path('pluscart/', views.plus_cart),
    path('minuscart/', views.minus_cart),
    path('removecart/', views.remove_cart),
    path('checkout/', views.checkout, name='checkout'),
    path('address/', views.address, name='address'),
    path('orders/', views.orders, name='orders'),
    path('paymentdone/', views.payment_done, name='paymentdone'),

    path('mobile/', views.mobile, name='mobile'),
    path('mobile/<slug:data>', views.mobile, name='mobiledata'),

    path('laptop/', views.laptop, name='laptop'),
    path('laptop/<slug:data>', views.laptop, name='laptopdata'),

    path('accounts/login/', auth_views.LoginView.as_view(template_name='app/login.html',
        authentication_form=LoginForm), name='login'),
```

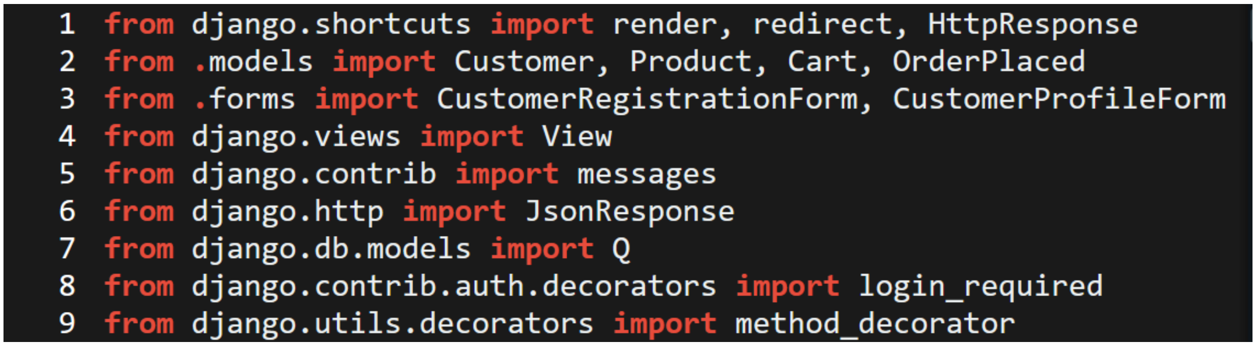


```
    # path('profile/', views.profile, name='profile'),
    path('profile/', views.ProfileView.as_view(), name='profile'),
    path('logout/', auth_views.LogoutView.as_view(next_page='login'), name='logout'),
    path('passwordchange/', auth_views.PasswordChangeView.as_view(template_name='app/
        passwordchange.html', form_class=MyPasswordChangeForm, success_url='/passwordchangedone/'),
        name='passwordchange'),
    path('passwordchangedone/', auth_views.PasswordChangeDoneView.as_view(template_name='app/
        passwordchangedone.html'), name='passwordchangedone'),

    path("password-reset/", auth_views.PasswordResetView.as_view(template_name='app/
        password_reset.html', form_class=MyPasswordResetForm), name="password_reset"),
    path("password-reset/done/", auth_views.PasswordResetDoneView.as_view(template_name='app/
        password_reset_done.html'), name="password_reset_done"),
    path("password-reset-confirm/<uidb64>/<token>/", auth_views.PasswordResetConfirmView.as_view(
        template_name='app/password_reset_confirm.html', form_class=MySetPasswordForm), name="
        password_reset_confirm"),
    path("password-reset-complete/", auth_views.PasswordResetCompleteView.as_view(template_name='app/
        password_reset_complete.html'), name="password_reset_complete"),

    path('registration/', views.CustomerRegistrationView.as_view(), name='customerregistration')
] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)

```

Here we have now set up all the links with their respective redirects.

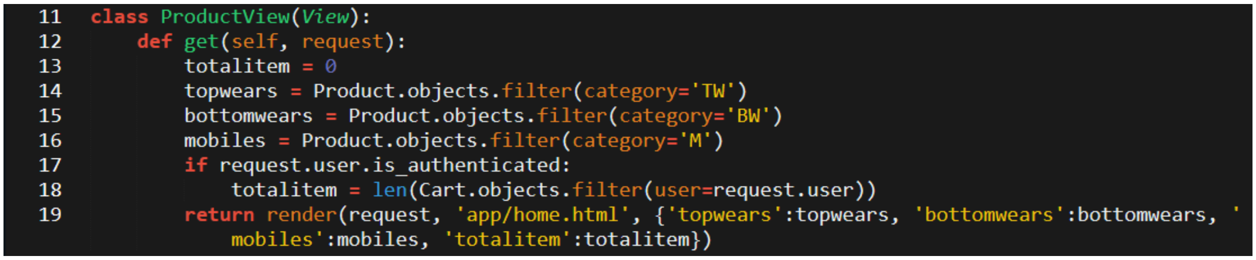
Editing views.py

Now we can work on views of our app, the most essential thing of our project so far. As always, first we will import everything. Start by coding this into the start portion of the **views.py** file.



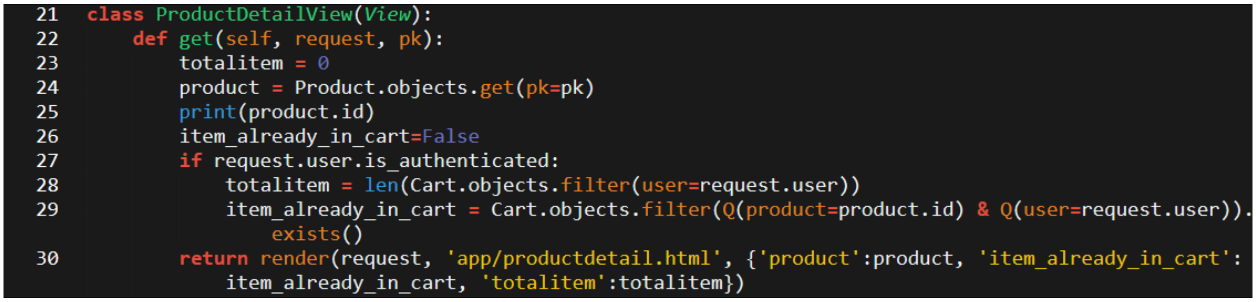
```
from django.shortcuts import render, redirect, HttpResponse
from .models import Customer, Product, Cart, OrderPlaced
from .forms import CustomerRegistrationForm, CustomerProfileForm
from django.views import View
from django.contrib import messages
from django.http import JsonResponse
from django.db.models import Q
from django.contrib.auth.decorators import login_required
from django.utils.decorators import method_decorator
```

Then we can code our **ProductView** view. Here we call **home.html** *(yet to be made)* in **return render** tag.



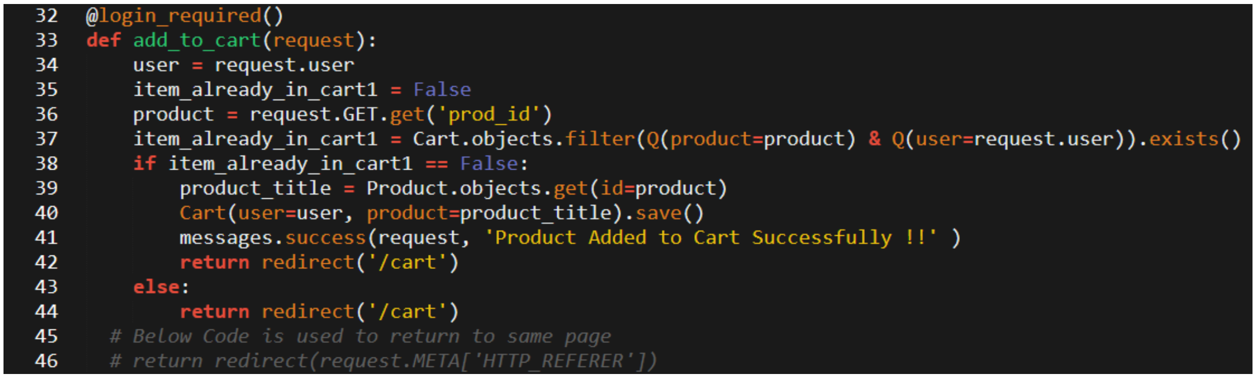
```
class ProductView(View):
    def get(self, request):
        totalitem = 0
        topwears = Product.objects.filter(category='TW')
        bottomwears = Product.objects.filter(category='BW')
        mobiles = Product.objects.filter(category='M')
        if request.user.is_authenticated:
            totalitem = len(Cart.objects.filter(user=request.user))
        return render(request, 'app/home.html', {'topwears':topwears, 'bottomwears':bottomwears, '
            mobiles':mobiles, 'totalitem':totalitem})
```

Then we will code the **ProductDetailView** view.



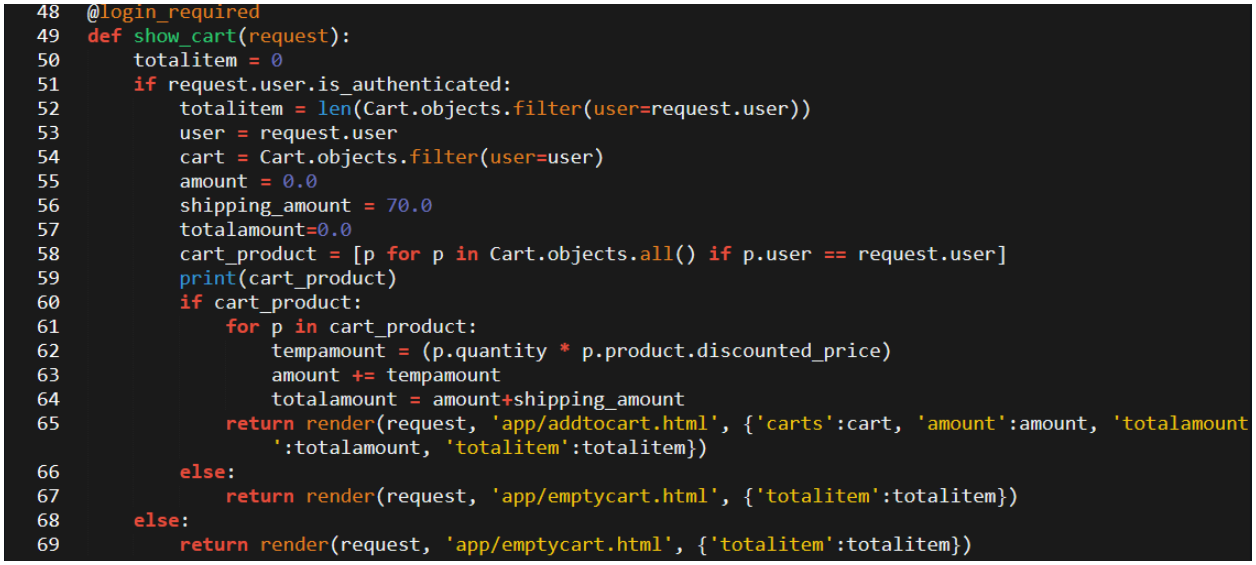
```
class ProductDetailView(View):
    def get(self, request, pk):
        totalitem = 0
        product = Product.objects.get(pk=pk)
        print(product.id)
        item_already_in_cart=False
        if request.user.is_authenticated:
            totalitem = len(Cart.objects.filter(user=request.user))
            item_already_in_cart = Cart.objects.filter(Q(product=product.id) & Q(user=request.user)).
                exists()
        return render(request, 'app/productdetail.html', {'product':product, 'item_already_in_cart':
            item_already_in_cart, 'totalitem':totalitem})
```

Now we code the **add_to_cart** view. Note that we have also included a **@login_required()** tag, which tells the program to run this code only when asked by the user and the user is logged in otherwise it will throw an error

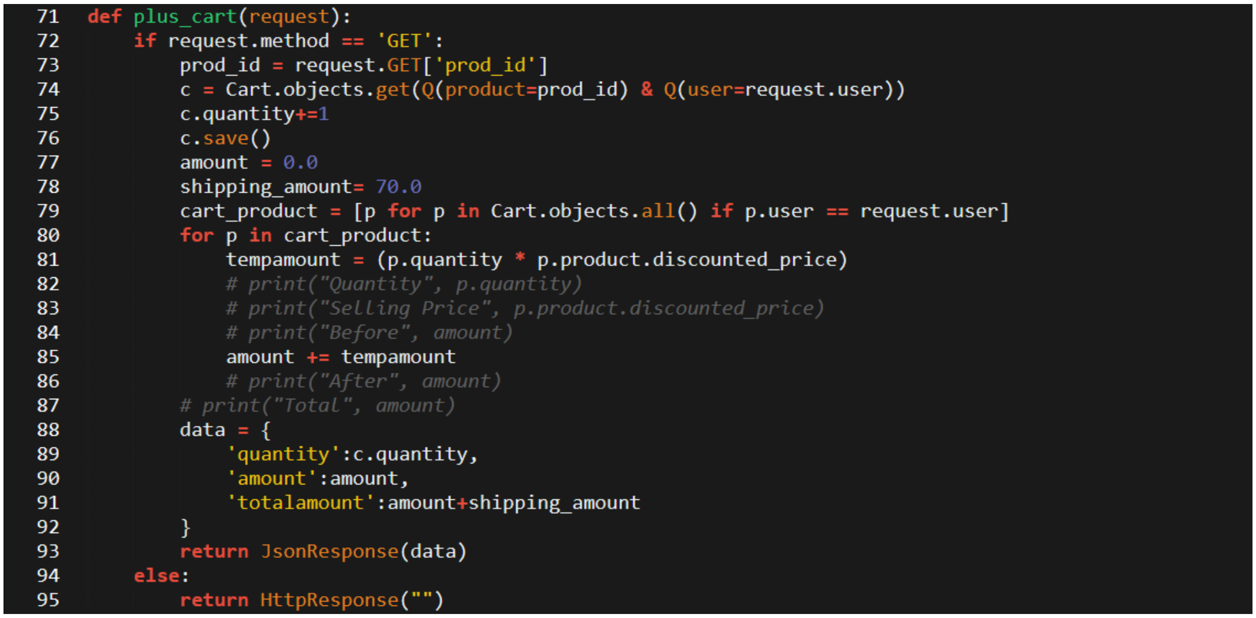


```
@login_required()
def add_to_cart(request):
    user = request.user
    item_already_in_cart1 = False
    product = request.GET.get('prod_id')
    item_already_in_cart1 = Cart.objects.filter(Q(product=product) & Q(user=request.user)).exists()
    if item_already_in_cart1 == False:
        product_title = Product.objects.get(id=product)
        Cart(user=user, product=product_title).save()
        messages.success(request, 'Product Added to Cart Successfully !!' )
        return redirect('/cart')
    else:
        return redirect('/cart')
  # Below Code is used to return to same page
  # return redirect(request.META['HTTP_REFERER'])
```

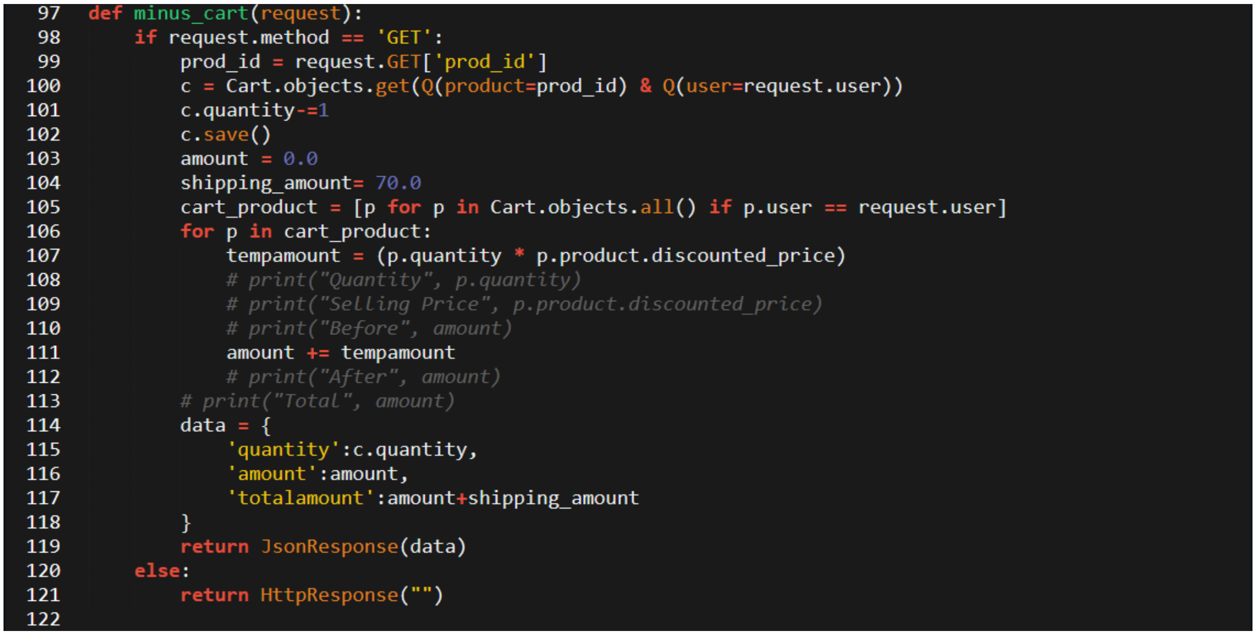
Then we create the following views step by step



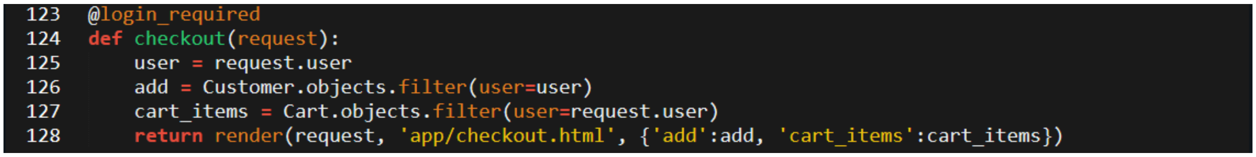
```python
@login_required
def show_cart(request):
    totalitem = 0
    if request.user.is_authenticated:
        totalitem = len(Cart.objects.filter(user=request.user))
        user = request.user
        cart = Cart.objects.filter(user=user)
        amount = 0.0
        shipping_amount = 70.0
        totalamount=0.0
        cart_product = [p for p in Cart.objects.all() if p.user == request.user]
        print(cart_product)
        if cart_product:
            for p in cart_product:
                tempamount = (p.quantity * p.product.discounted_price)
                amount += tempamount
                totalamount = amount+shipping_amount
            return render(request, 'app/addtocart.html', {'carts':cart, 'amount':amount, 'totalamount
                ':totalamount, 'totalitem':totalitem})
        else:
            return render(request, 'app/emptycart.html', {'totalitem':totalitem})
    else:
        return render(request, 'app/emptycart.html', {'totalitem':totalitem})
```



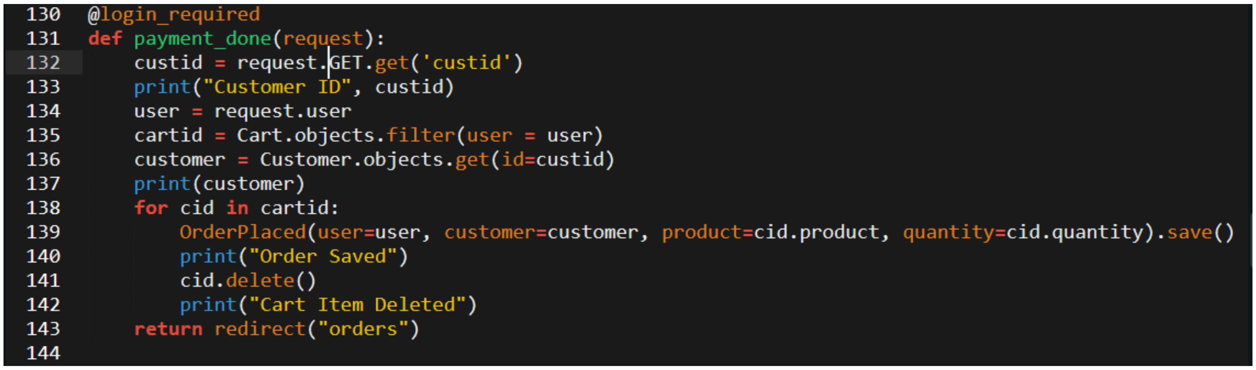
```python
def plus_cart(request):
    if request.method == 'GET':
        prod_id = request.GET['prod_id']
        c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
        c.quantity+=1
        c.save()
        amount = 0.0
        shipping_amount= 70.0
        cart_product = [p for p in Cart.objects.all() if p.user == request.user]
        for p in cart_product:
            tempamount = (p.quantity * p.product.discounted_price)
            # print("Quantity", p.quantity)
            # print("Selling Price", p.product.discounted_price)
            # print("Before", amount)
            amount += tempamount
            # print("After", amount)
        # print("Total", amount)
        data = {
            'quantity':c.quantity,
            'amount':amount,
            'totalamount':amount+shipping_amount
        }
        return JsonResponse(data)
    else:
        return HttpResponse("")
```



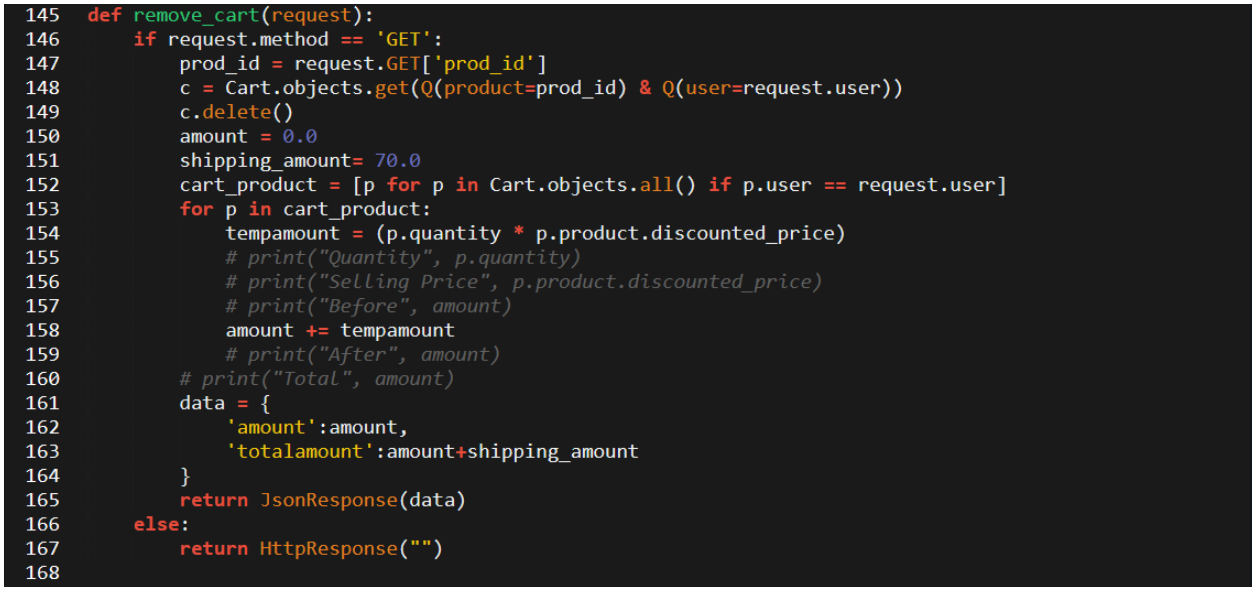
```python
def minus_cart(request):
    if request.method == 'GET':
        prod_id = request.GET['prod_id']
        c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
        c.quantity-=1
        c.save()
        amount = 0.0
        shipping_amount= 70.0
        cart_product = [p for p in Cart.objects.all() if p.user == request.user]
        for p in cart_product:
            tempamount = (p.quantity * p.product.discounted_price)
            # print("Quantity", p.quantity)
            # print("Selling Price", p.product.discounted_price)
            # print("Before", amount)
            amount += tempamount
            # print("After", amount)
        # print("Total", amount)
        data = {
            'quantity':c.quantity,
            'amount':amount,
            'totalamount':amount+shipping_amount
        }
        return JsonResponse(data)
    else:
        return HttpResponse("")

```



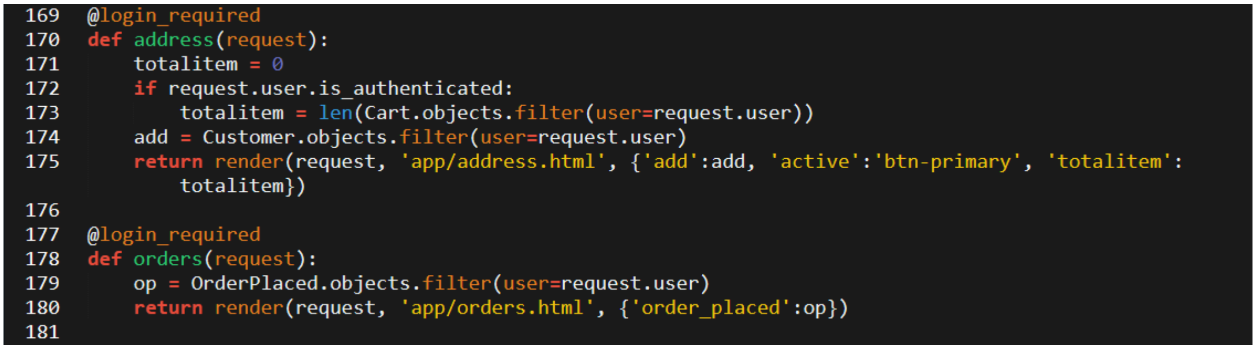
```python
@login_required
def checkout(request):
    user = request.user
    add = Customer.objects.filter(user=user)
    cart_items = Cart.objects.filter(user=request.user)
    return render(request, 'app/checkout.html', {'add':add, 'cart_items':cart_items})
```



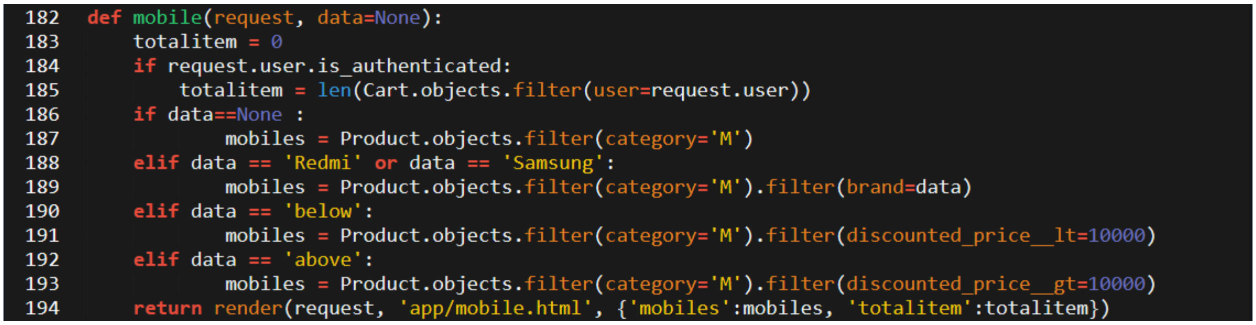
```python
@login_required
def payment_done(request):
    custid = request.GET.get('custid')
    print("Customer ID", custid)
    user = request.user
    cartid = Cart.objects.filter(user = user)
    customer = Customer.objects.get(id=custid)
    print(customer)
    for cid in cartid:
        OrderPlaced(user=user, customer=customer, product=cid.product, quantity=cid.quantity).save()
        print("Order Saved")
        cid.delete()
        print("Cart Item Deleted")
    return redirect("orders")

```



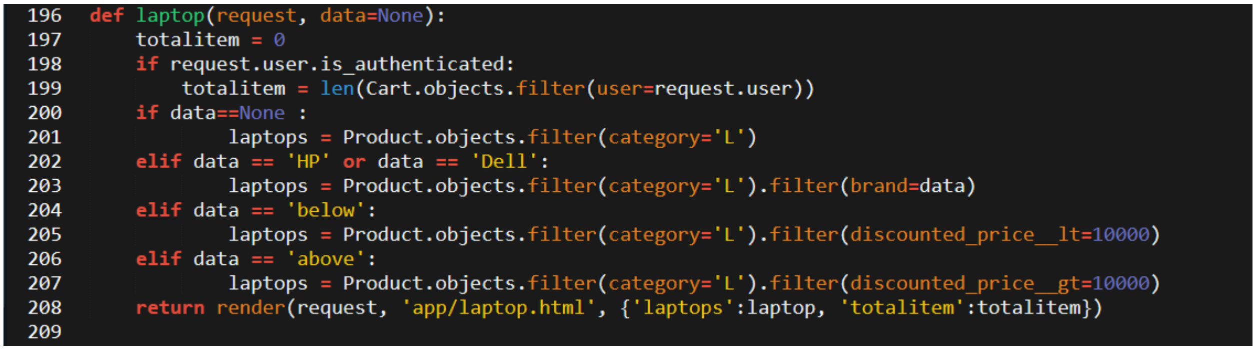
```python
def remove_cart(request):
    if request.method == 'GET':
        prod_id = request.GET['prod_id']
        c = Cart.objects.get(Q(product=prod_id) & Q(user=request.user))
        c.delete()
        amount = 0.0
        shipping_amount= 70.0
        cart_product = [p for p in Cart.objects.all() if p.user == request.user]
        for p in cart_product:
            tempamount = (p.quantity * p.product.discounted_price)
            # print("Quantity", p.quantity)
            # print("Selling Price", p.product.discounted_price)
            # print("Before", amount)
            amount += tempamount
            # print("After", amount)
        # print("Total", amount)
        data = {
            'amount':amount,
            'totalamount':amount+shipping_amount
        }
        return JsonResponse(data)
    else:
        return HttpResponse("")

```



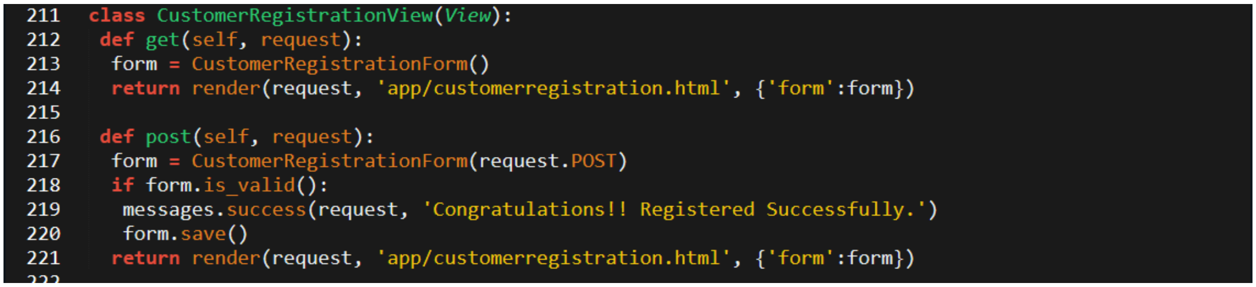
```python
@login_required
def address(request):
    totalitem = 0
    if request.user.is_authenticated:
        totalitem = len(Cart.objects.filter(user=request.user))
    add = Customer.objects.filter(user=request.user)
    return render(request, 'app/address.html', {'add':add, 'active':'btn-primary', 'totalitem':totalitem})

@login_required
def orders(request):
    op = OrderPlaced.objects.filter(user=request.user)
    return render(request, 'app/orders.html', {'order_placed':op})

```



```python
def mobile(request, data=None):
    totalitem = 0
    if request.user.is_authenticated:
        totalitem = len(Cart.objects.filter(user=request.user))
    if data==None :
            mobiles = Product.objects.filter(category='M')
    elif data == 'Redmi' or data == 'Samsung':
            mobiles = Product.objects.filter(category='M').filter(brand=data)
    elif data == 'below':
            mobiles = Product.objects.filter(category='M').filter(discounted_price__lt=10000)
    elif data == 'above':
            mobiles = Product.objects.filter(category='M').filter(discounted_price__gt=10000)
    return render(request, 'app/mobile.html', {'mobiles':mobiles, 'totalitem':totalitem})
```



```python
def laptop(request, data=None):
    totalitem = 0
    if request.user.is_authenticated:
        totalitem = len(Cart.objects.filter(user=request.user))
    if data==None :
            laptops = Product.objects.filter(category='L')
    elif data == 'HP' or data == 'Dell':
            laptops = Product.objects.filter(category='L').filter(brand=data)
    elif data == 'below':
            laptops = Product.objects.filter(category='L').filter(discounted_price__lt=10000)
    elif data == 'above':
            laptops = Product.objects.filter(category='L').filter(discounted_price__gt=10000)
    return render(request, 'app/laptop.html', {'laptops':laptop, 'totalitem':totalitem})

```

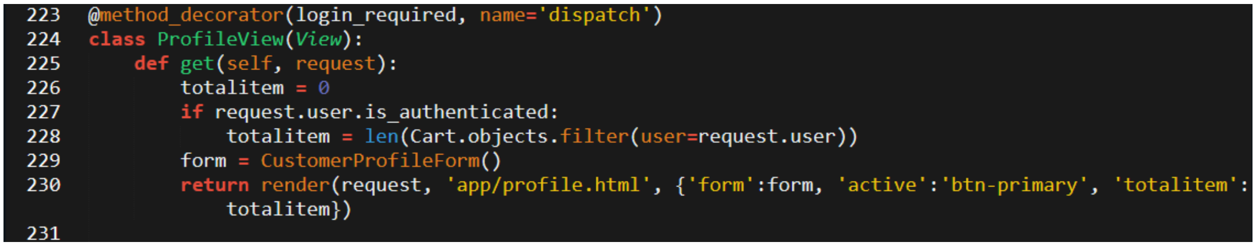


```python
class CustomerRegistrationView(View):
 def get(self, request):
  form = CustomerRegistrationForm()
  return render(request, 'app/customerregistration.html', {'form':form})

 def post(self, request):
  form = CustomerRegistrationForm(request.POST)
  if form.is_valid():
   messages.success(request, 'Congratulations!! Registered Successfully.')
   form.save()
  return render(request, 'app/customerregistration.html', {'form':form})
```

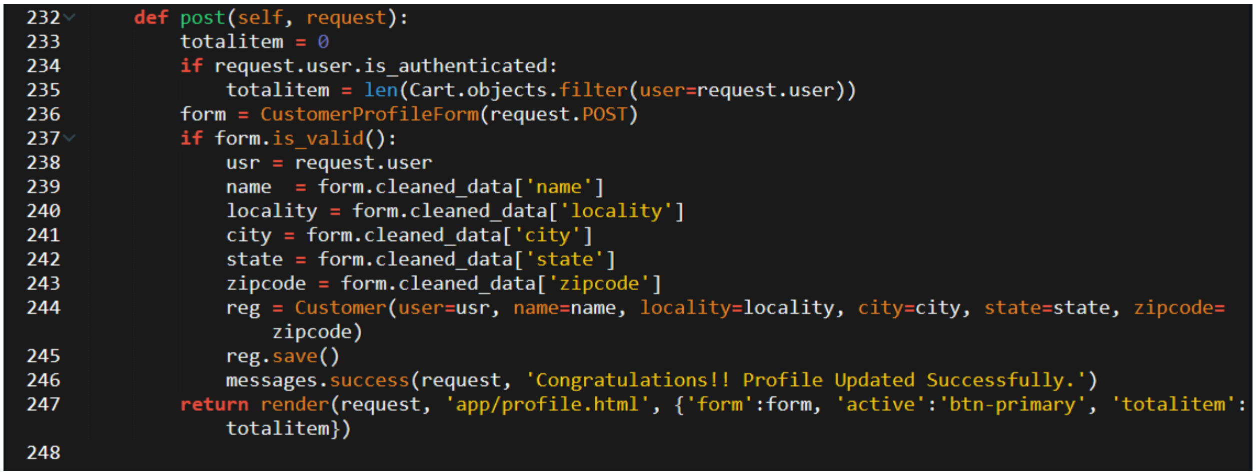
Note that here we add a new tag called **@method_decorator.**

To decorate every instance of a class-based view, you need to decorate the class definition itself. To do this you apply the decorator to the dispatch() method of the class.

A method on a class isn't quite the same as a standalone function, so you can't just apply a function decorator to the method – you need to transform it into a method decorator first. The **method_decorator** decorator transforms a function decorator into a method decorator so that it can be used on an instance method. More info at docs@django



```python
@method_decorator(login_required, name='dispatch')
class ProfileView(View):
    def get(self, request):
        totalitem = 0
        if request.user.is_authenticated:
            totalitem = len(Cart.objects.filter(user=request.user))
        form = CustomerProfileForm()
        return render(request, 'app/profile.html', {'form':form, 'active':'btn-primary', 'totalitem':
            totalitem})

```



```python
    def post(self, request):
        totalitem = 0
        if request.user.is_authenticated:
            totalitem = len(Cart.objects.filter(user=request.user))
        form = CustomerProfileForm(request.POST)
        if form.is_valid():
            usr = request.user
            name  = form.cleaned_data['name']
            locality = form.cleaned_data['locality']
            city = form.cleaned_data['city']
            state = form.cleaned_data['state']
            zipcode = form.cleaned_data['zipcode']
            reg = Customer(user=usr, name=name, locality=locality, city=city, state=state, zipcode=
                zipcode)
            reg.save()
            messages.success(request, 'Congratulations!! Profile Updated Successfully.')
        return render(request, 'app/profile.html', {'form':form, 'active':'btn-primary', 'totalitem':
            totalitem})

```

Now before moving any further, I'd recommend you to apply your migrations. To do so, open a cmd window at your **manage.py** location and run **py manage.py migrate**. This will make sure that there are no issues with any migrations while running the project. Now running the runserver command will display an error as we haven't yet created the HTML files. So, let's just do that exact thing!

Step 4 | Creating HTML files

Coding base.html

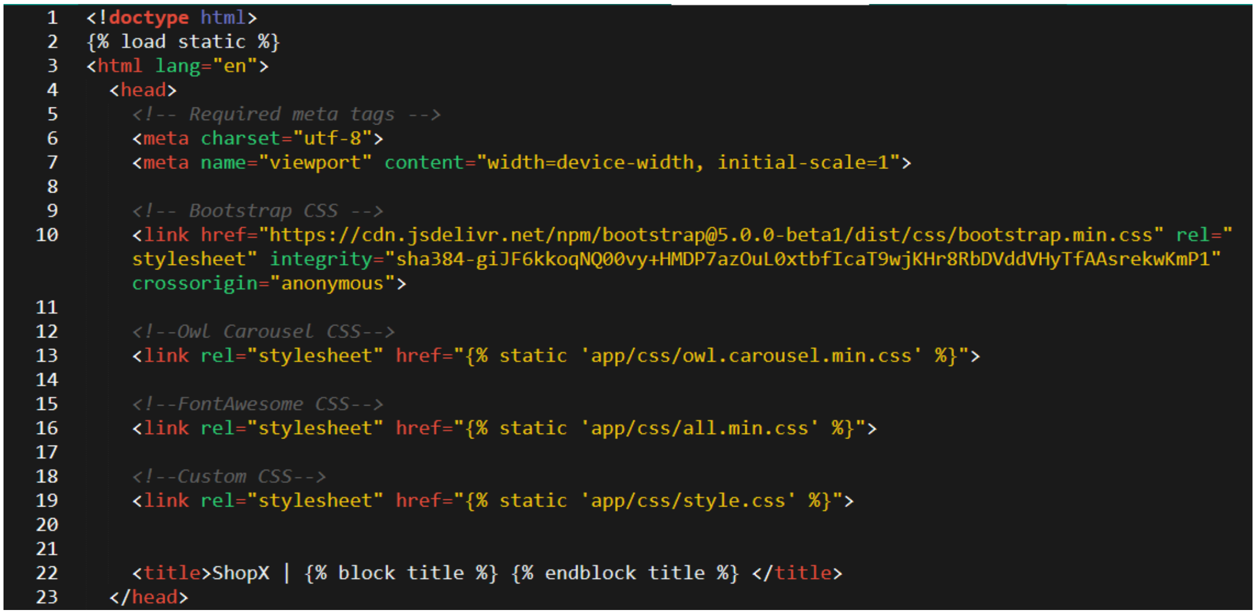
Now that we have done all of our backend *(I hope so)*, let's work on our front-end!

First off, we need to create base.html in the templates folder inside the app folder. Including other HTML files mentioned in this documentation, the file structure should look something like this. Feel free to create these files beforehand as that will save some of your precious seconds.

```
.
└── templates
    └── app
        ├── address.html
        ├── addtocart.html
        ├── base.html <- The file we are going to create first
        ├── checkout.html
        ├── customerregistration.html
        ├── emptycart.html
        ├── home.html
        ├── laptop.html
        ├── mobile.html
        ├── orders.html
        ├── password_reset.html
        ├── password_reset_complete.html
        ├── password_reset_confirm.html
        ├── password_reset_done.html
        ├── passwordchange.html
        ├── passwordchangedone.html
        ├── productdetail.html
        └── profile.html
```

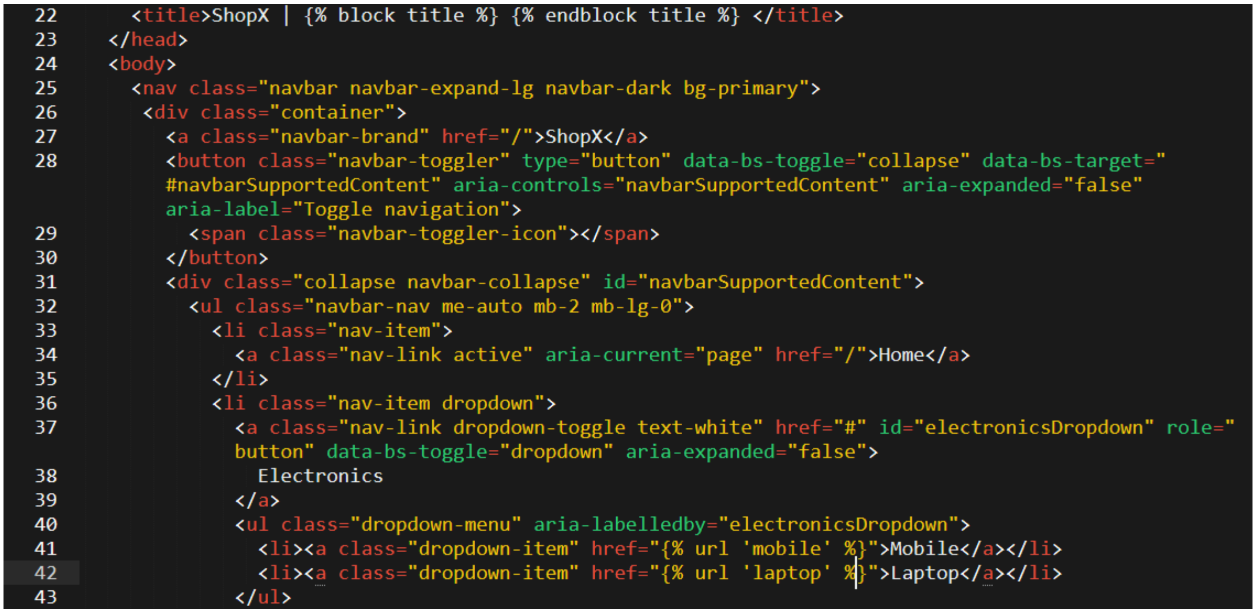
Now we will begin with base.html

Here we load the static URL defined in the urls.py file to utilize the static files. Here we also complete our <head> part of our base.html.

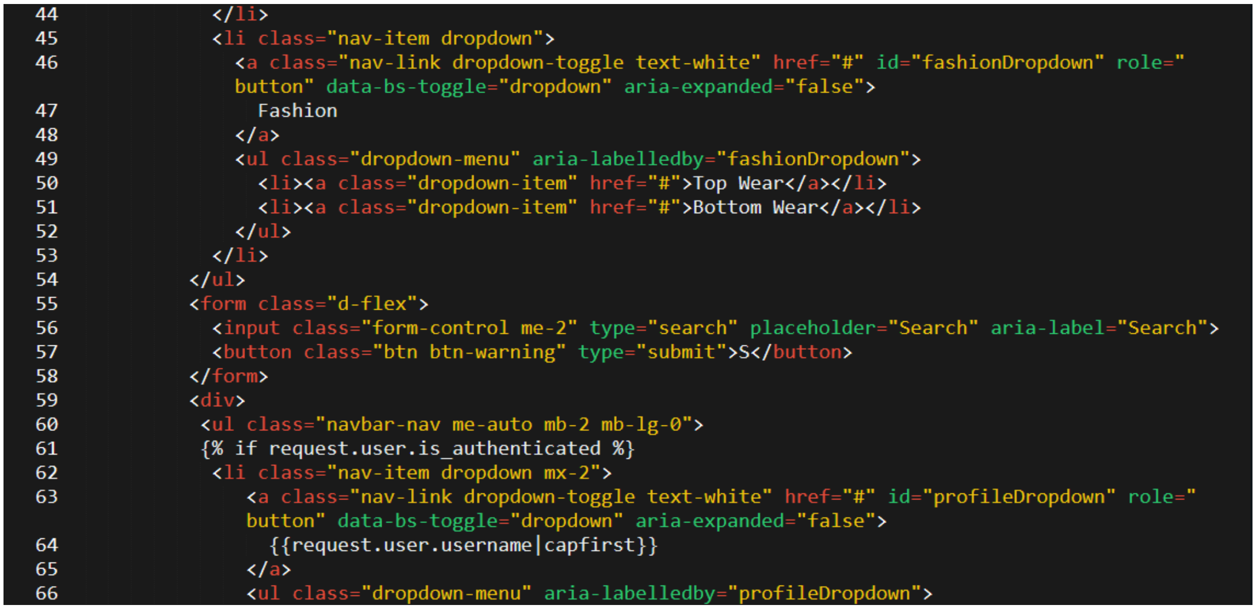


```html
<!doctype html>
{% load static %}
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-giJF6kkoqNQ00vy+HMDP7azOuL0xtbfIcaT9wjKHr8RbDVddVHyTfAAsrekwKmP1" crossorigin="anonymous">

    <!--Owl Carousel CSS-->
    <link rel="stylesheet" href="{% static 'app/css/owl.carousel.min.css' %}">

    <!--FontAwesome CSS-->
    <link rel="stylesheet" href="{% static 'app/css/all.min.css' %}">

    <!--Custom CSS-->
    <link rel="stylesheet" href="{% static 'app/css/style.css' %}">


    <title>ShopX | {% block title %} {% endblock title %} </title>
  </head>
```

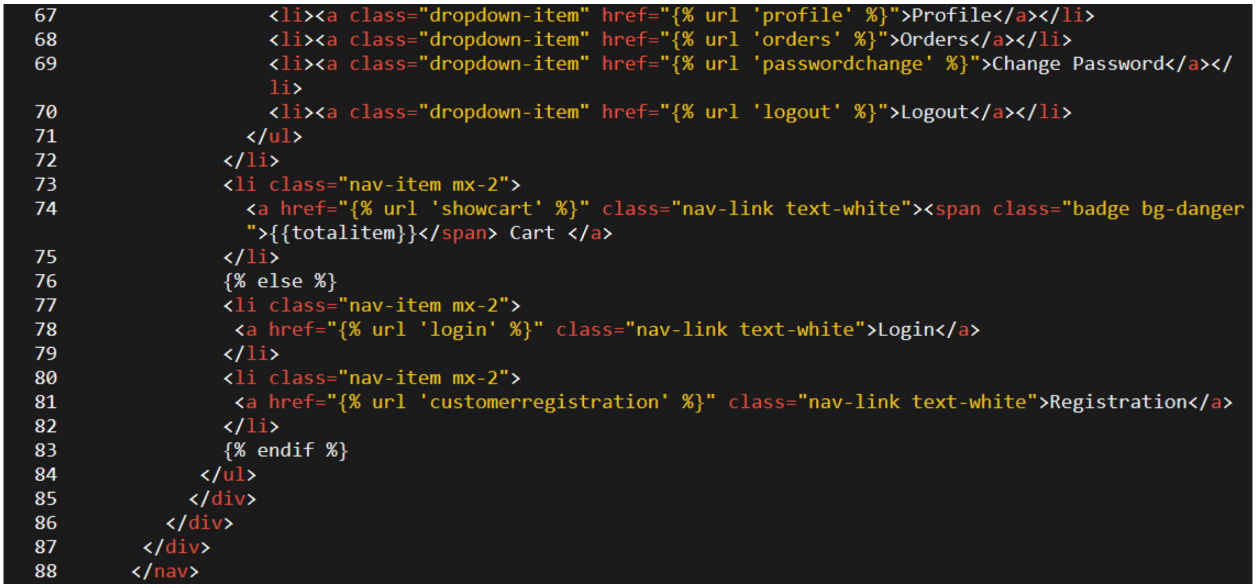
Then we code the remaining part of our file as follows



```
      <title>ShopX | {% block title %} {% endblock title %} </title>
    </head>
    <body>
      <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
       <div class="container">
         <a class="navbar-brand" href="/">ShopX</a>
         <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
           <span class="navbar-toggler-icon"></span>
         </button>
         <div class="collapse navbar-collapse" id="navbarSupportedContent">
           <ul class="navbar-nav me-auto mb-2 mb-lg-0">
             <li class="nav-item">
               <a class="nav-link active" aria-current="page" href="/">Home</a>
             </li>
             <li class="nav-item dropdown">
               <a class="nav-link dropdown-toggle text-white" href="#" id="electronicsDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">
                 Electronics
               </a>
               <ul class="dropdown-menu" aria-labelledby="electronicsDropdown">
                 <li><a class="dropdown-item" href="{% url 'mobile' %}">Mobile</a></li>
                 <li><a class="dropdown-item" href="{% url 'laptop' %}">Laptop</a></li>
               </ul>
```

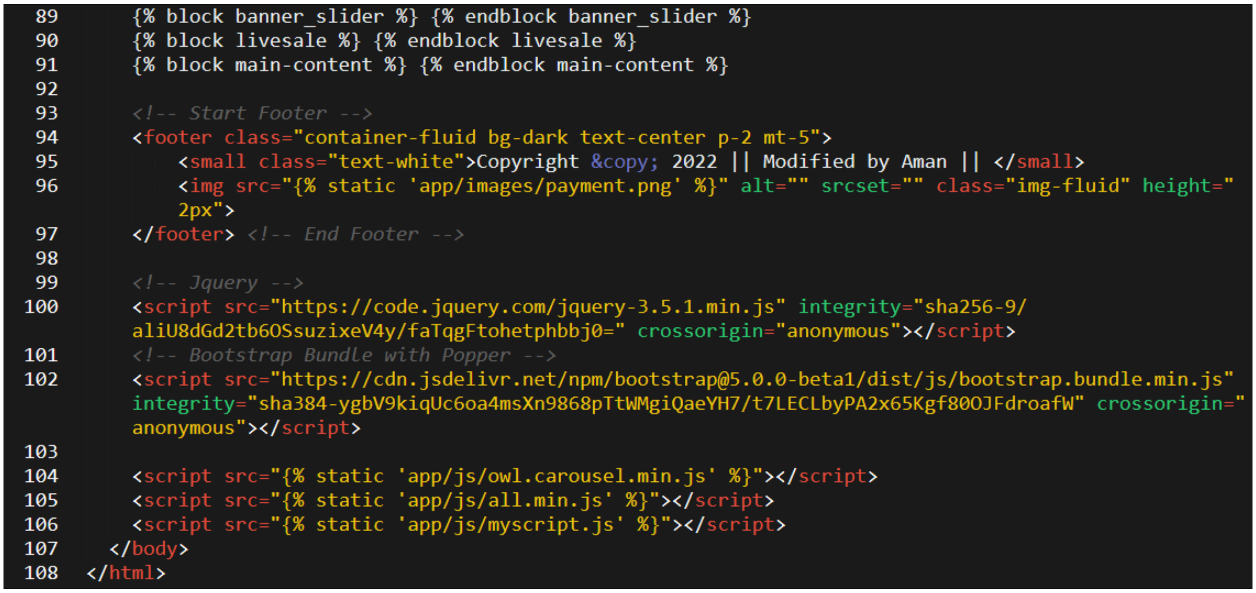


```
             </li>
             <li class="nav-item dropdown">
               <a class="nav-link dropdown-toggle text-white" href="#" id="fashionDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">
                 Fashion
               </a>
               <ul class="dropdown-menu" aria-labelledby="fashionDropdown">
                 <li><a class="dropdown-item" href="#">Top Wear</a></li>
                 <li><a class="dropdown-item" href="#">Bottom Wear</a></li>
               </ul>
             </li>
           </ul>
           <form class="d-flex">
             <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
             <button class="btn btn-warning" type="submit">S</button>
           </form>
           <div>
            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
            {% if request.user.is_authenticated %}
             <li class="nav-item dropdown mx-2">
                <a class="nav-link dropdown-toggle text-white" href="#" id="profileDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">
                  {{request.user.username|capfirst}}
                </a>
                <ul class="dropdown-menu" aria-labelledby="profileDropdown">
```



```
                <li><a class="dropdown-item" href="{% url 'profile' %}">Profile</a></li>
                <li><a class="dropdown-item" href="{% url 'orders' %}">Orders</a></li>
                <li><a class="dropdown-item" href="{% url 'passwordchange' %}">Change Password</a></li>
                <li><a class="dropdown-item" href="{% url 'logout' %}">Logout</a></li>
              </ul>
            </li>
            <li class="nav-item mx-2">
              <a href="{% url 'showcart' %}" class="nav-link text-white"><span class="badge bg-danger">{{totalitem}}</span> Cart </a>
            </li>
            {% else %}
            <li class="nav-item mx-2">
             <a href="{% url 'login' %}" class="nav-link text-white">Login</a>
            </li>
            <li class="nav-item mx-2">
             <a href="{% url 'customerregistration' %}" class="nav-link text-white">Registration</a>
            </li>
            {% endif %}
          </ul>
         </div>
       </div>
     </div>
    </nav>
```

Now we add our footer to our base.html



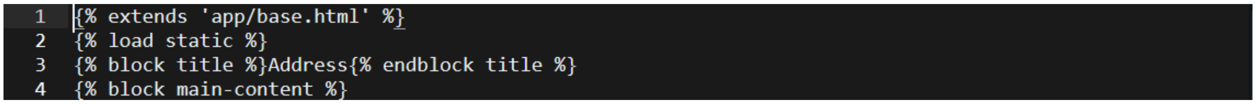
```
    {% block banner_slider %} {% endblock banner_slider %}
    {% block livesale %} {% endblock livesale %}
    {% block main-content %} {% endblock main-content %}

    <!-- Start Footer -->
    <footer class="container-fluid bg-dark text-center p-2 mt-5">
        <small class="text-white">Copyright &copy; 2022 || Modified by Aman || </small>
        <img src="{% static 'app/images/payment.png' %}" alt="" srcset="" class="img-fluid" height="2px">
    </footer> <!-- End Footer -->

    <!-- Jquery -->
    <script src="https://code.jquery.com/jquery-3.5.1.min.js" integrity="sha256-9/aliU8dGd2tb6OSsuzixeV4y/faTqgFtohetphbbj0=" crossorigin="anonymous"></script>
    <!-- Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta1/dist/js/bootstrap.bundle.min.js" integrity="sha384-ygbV9kiqUc6oa4msXn9868pTtWMgiQaeYH7/t7LECLbyPA2x65Kgf80OJFdroafW" crossorigin="anonymous"></script>

    <script src="{% static 'app/js/owl.carousel.min.js' %}"></script>
    <script src="{% static 'app/js/all.min.js' %}"></script>
    <script src="{% static 'app/js/myscript.js' %}"></script>
  </body>
</html>
```

Coding other .html files

address.html

Here you can see, that we have included a special tag

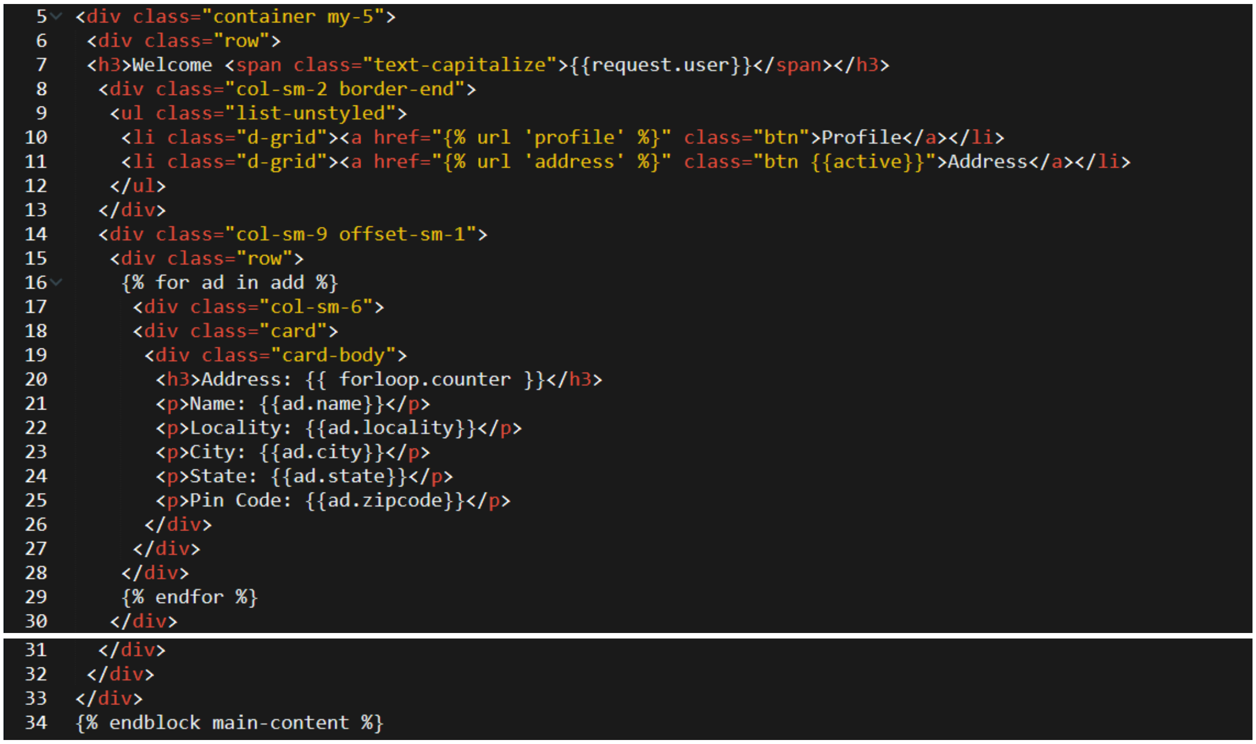
{% extends ‘app/base.html’ %}



```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Address{% endblock title %}
{% block main-content %}
```

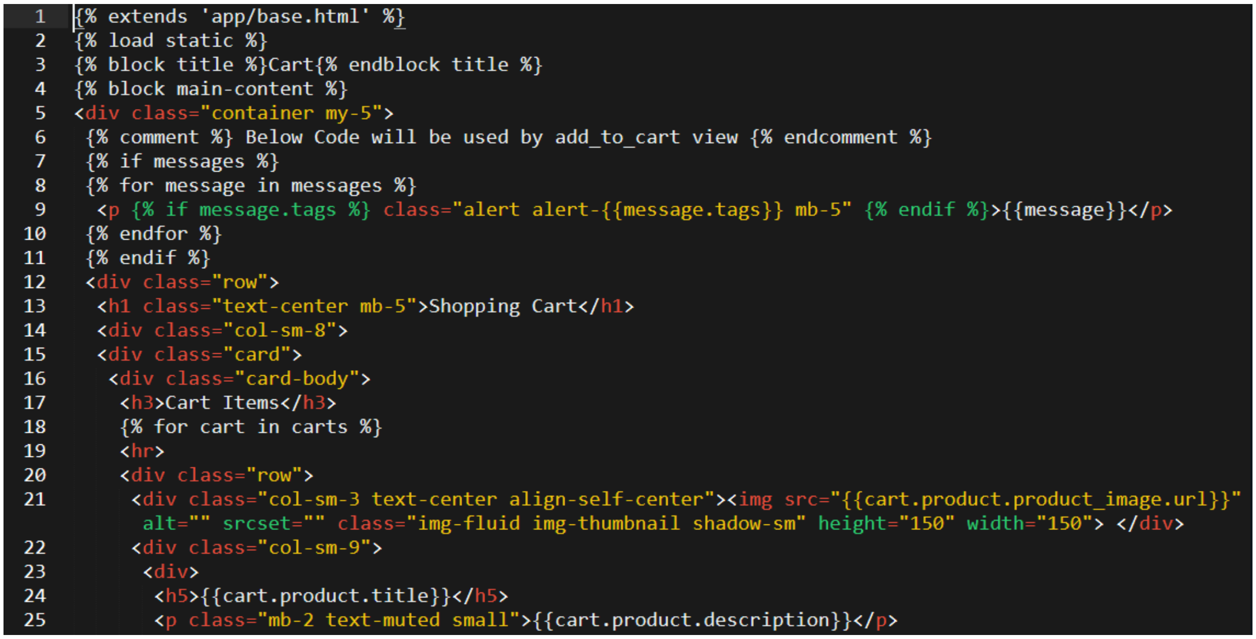
What this tag does is it loads our base.html as … well our base code and adds the following code after <head> block.

Now we add the remaining code to the file

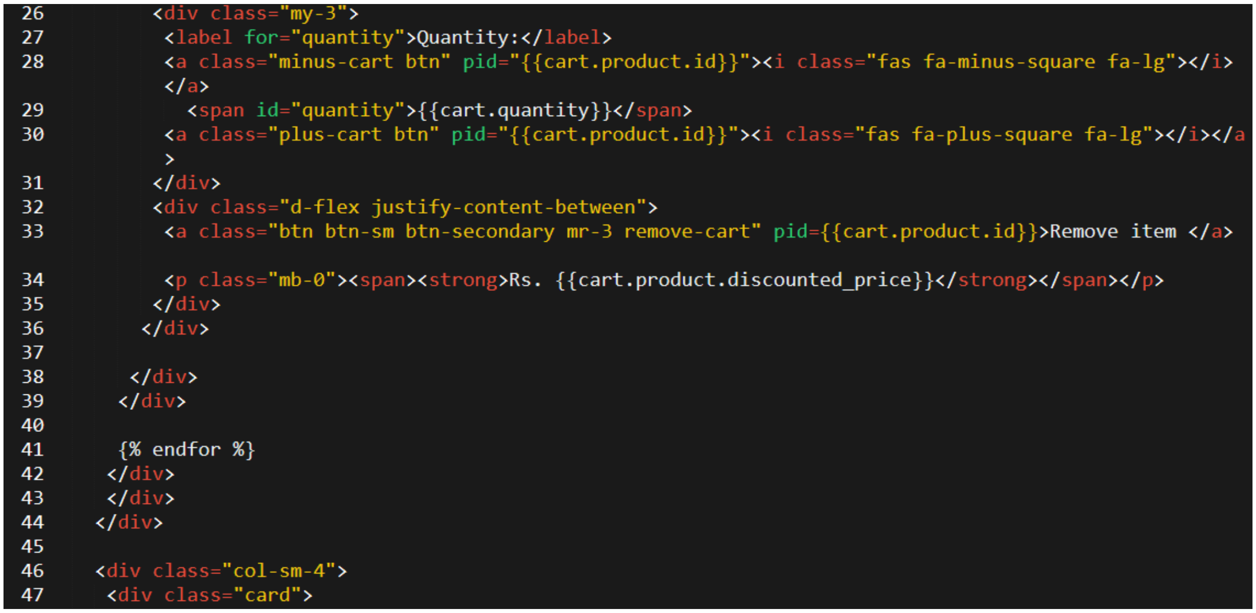


```
<div class="container my-5">
 <div class="row">
 <h3>Welcome <span class="text-capitalize">{{request.user}}</span></h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'profile' %}" class="btn">Profile</a></li>
    <li class="d-grid"><a href="{% url 'address' %}" class="btn {{active}}">Address</a></li>
   </ul>
  </div>
  <div class="col-sm-9 offset-sm-1">
   <div class="row">
    {% for ad in add %}
     <div class="col-sm-6">
     <div class="card">
      <div class="card-body">
       <h3>Address: {{ forloop.counter }}</h3>
       <p>Name: {{ad.name}}</p>
       <p>Locality: {{ad.locality}}</p>
       <p>City: {{ad.city}}</p>
       <p>State: {{ad.state}}</p>
       <p>Pin Code: {{ad.zipcode}}</p>
      </div>
     </div>
    </div>
    {% endfor %}
   </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

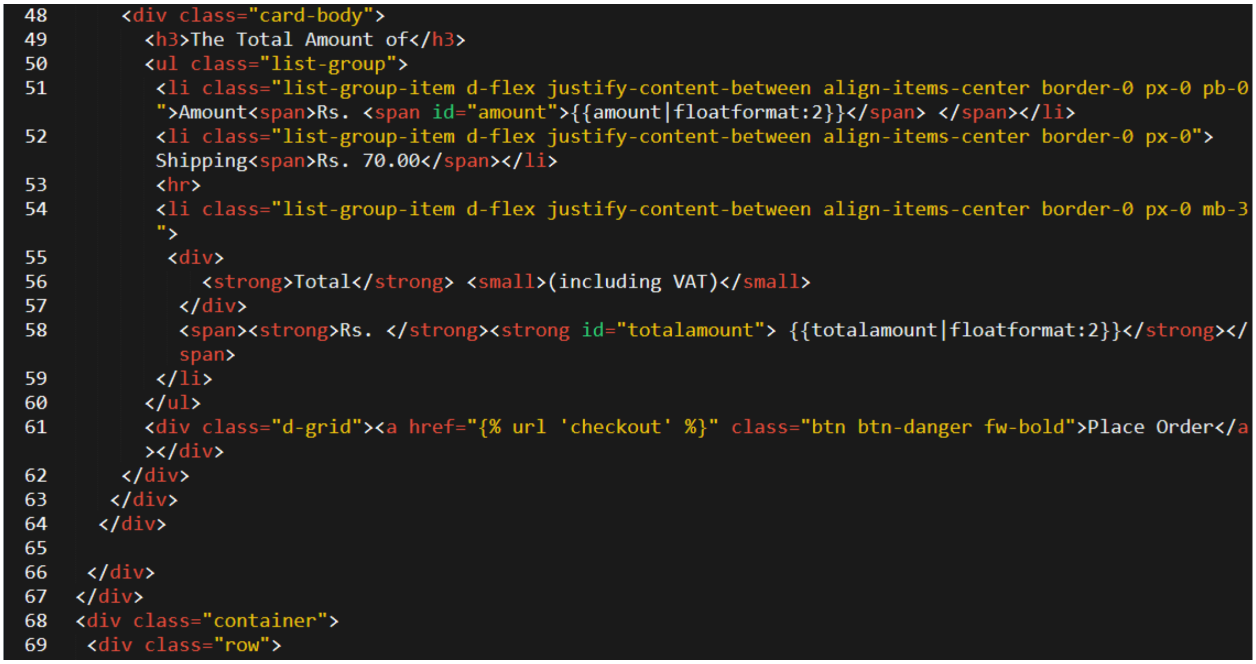
addtocart.html



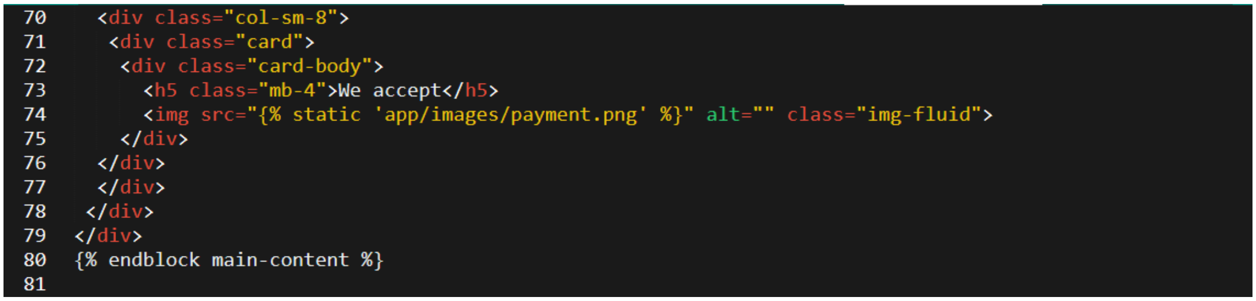
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Cart{% endblock title %}
{% block main-content %}
<div class="container my-5">
 {% comment %} Below Code will be used by add_to_cart view {% endcomment %}
 {% if messages %}
 {% for message in messages %}
  <p {% if message.tags %} class="alert alert-{{message.tags}} mb-5" {% endif %}>{{message}}</p>
 {% endfor %}
 {% endif %}
 <div class="row">
  <h1 class="text-center mb-5">Shopping Cart</h1>
  <div class="col-sm-8">
  <div class="card">
   <div class="card-body">
    <h3>Cart Items</h3>
    {% for cart in carts %}
    <hr>
    <div class="row">
     <div class="col-sm-3 text-center align-self-center"><img src="{{cart.product.product_image.url}}" alt="" srcset="" class="img-fluid img-thumbnail shadow-sm" height="150" width="150"> </div>
     <div class="col-sm-9">
      <div>
       <h5>{{cart.product.title}}</h5>
       <p class="mb-2 text-muted small">{{cart.product.description}}</p>
```



```
       <div class="my-3">
        <label for="quantity">Quantity:</label>
        <a class="minus-cart btn" pid="{{cart.product.id}}"><i class="fas fa-minus-square fa-lg"></i></a>
          <span id="quantity">{{cart.quantity}}</span>
        <a class="plus-cart btn" pid="{{cart.product.id}}"><i class="fas fa-plus-square fa-lg"></i></a>
       </div>
       <div class="d-flex justify-content-between">
        <a class="btn btn-sm btn-secondary mr-3 remove-cart" pid={{cart.product.id}}>Remove item </a>

        <p class="mb-0"><span><strong>Rs. {{cart.product.discounted_price}}</strong></span></p>
       </div>
      </div>

     </div>
    </div>

    {% endfor %}
   </div>
   </div>
  </div>

  <div class="col-sm-4">
   <div class="card">
```

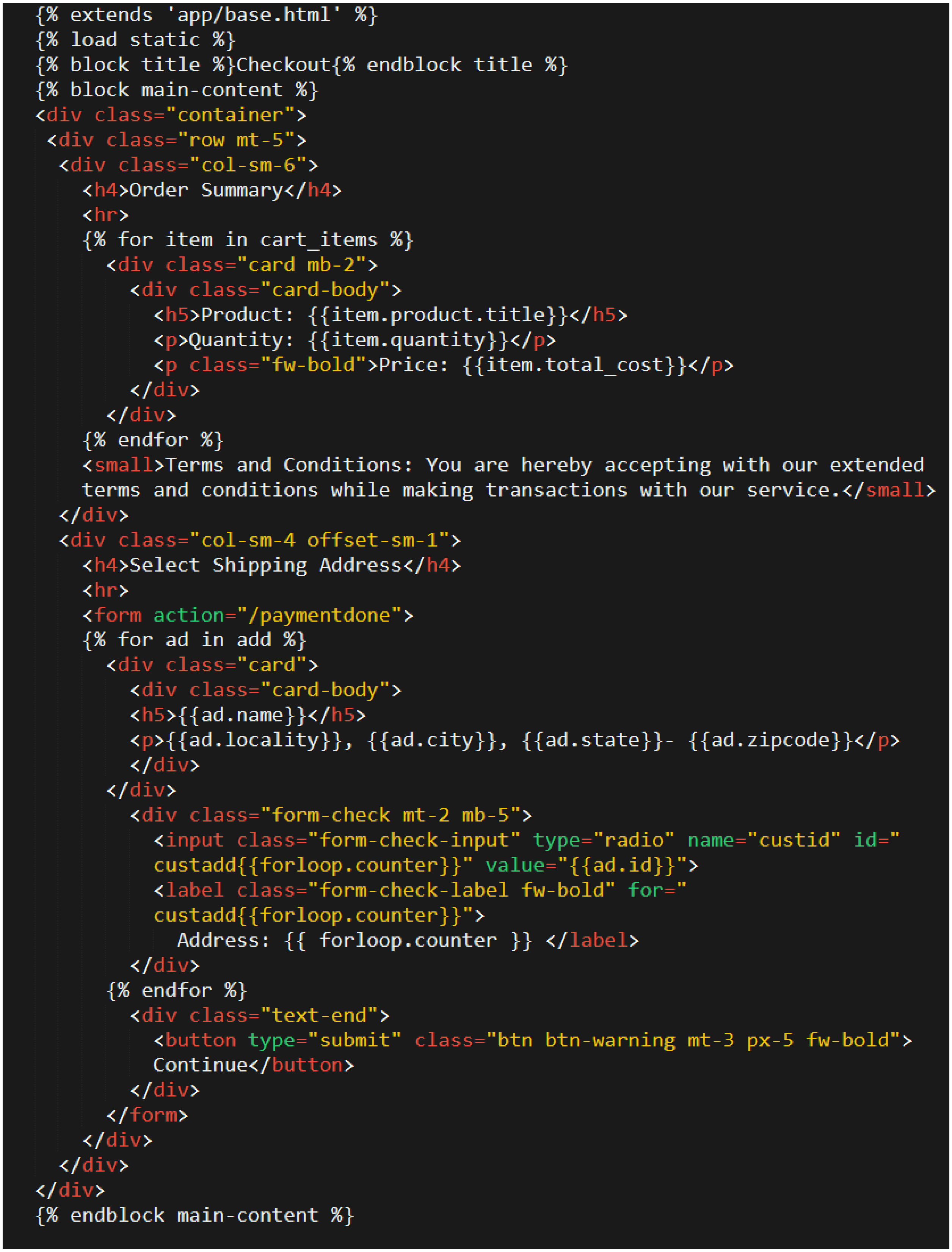


```
    <div class="card-body">
      <h3>The Total Amount of</h3>
      <ul class="list-group">
       <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-0 pb-0">Amount<span>Rs. <span id="amount">{{amount|floatformat:2}}</span> </span></li>
       <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-0">Shipping<span>Rs. 70.00</span></li>
       <hr>
       <li class="list-group-item d-flex justify-content-between align-items-center border-0 px-0 mb-3">
        <div>
           <strong>Total</strong> <small>(including VAT)</small>
         </div>
         <span><strong>Rs. </strong><strong id="totalamount"> {{totalamount|floatformat:2}}</strong></span>
       </li>
      </ul>
      <div class="d-grid"><a href="{% url 'checkout' %}" class="btn btn-danger fw-bold">Place Order</a></div>
    </div>
   </div>
  </div>

 </div>
</div>
<div class="container">
 <div class="row">
```



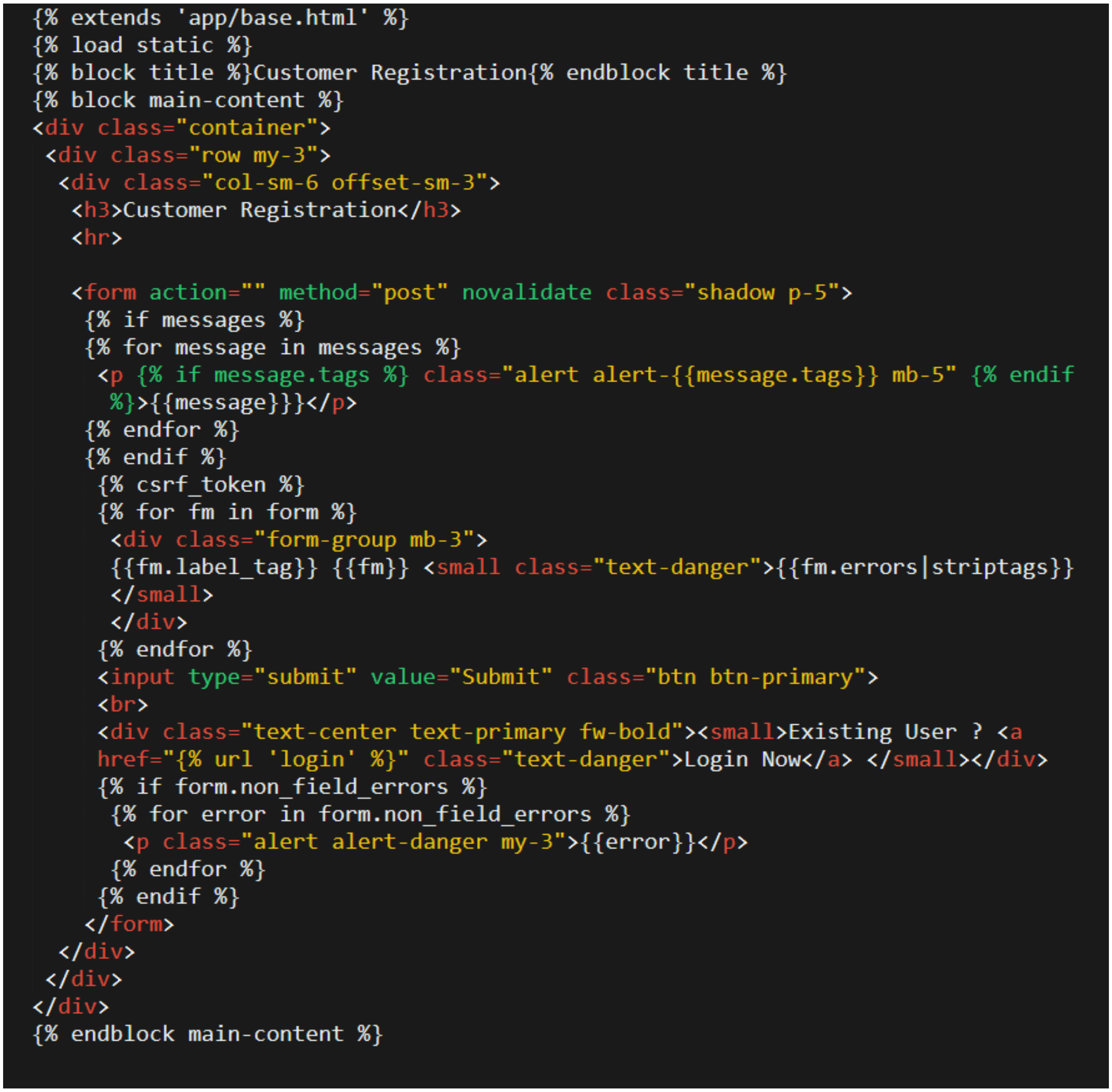
```
  <div class="col-sm-8">
   <div class="card">
    <div class="card-body">
      <h5 class="mb-4">We accept</h5>
      <img src="{% static 'app/images/payment.png' %}" alt="" class="img-fluid">
    </div>
  </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

checkout.html



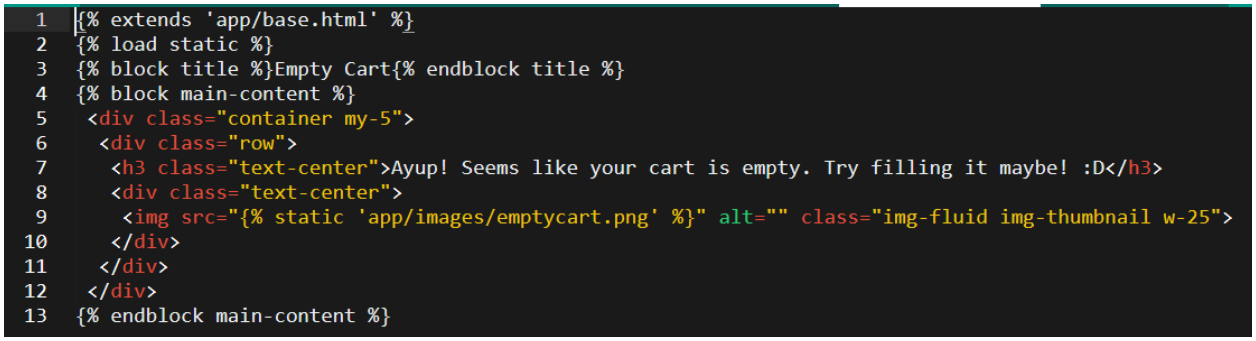
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Checkout{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row mt-5">
  <div class="col-sm-6">
    <h4>Order Summary</h4>
    <hr>
    {% for item in cart_items %}
      <div class="card mb-2">
        <div class="card-body">
          <h5>Product: {{item.product.title}}</h5>
          <p>Quantity: {{item.quantity}}</p>
          <p class="fw-bold">Price: {{item.total_cost}}</p>
        </div>
      </div>
    {% endfor %}
    <small>Terms and Conditions: You are hereby accepting with our extended
    terms and conditions while making transactions with our service.</small>
  </div>
  <div class="col-sm-4 offset-sm-1">
    <h4>Select Shipping Address</h4>
    <hr>
    <form action="/paymentdone">
    {% for ad in add %}
      <div class="card">
        <div class="card-body">
        <h5>{{ad.name}}</h5>
        <p>{{ad.locality}}, {{ad.city}}, {{ad.state}}- {{ad.zipcode}}</p>
        </div>
      </div>
        <div class="form-check mt-2 mb-5">
          <input class="form-check-input" type="radio" name="custid" id="
          custadd{{forloop.counter}}" value="{{ad.id}}">
          <label class="form-check-label fw-bold" for="
          custadd{{forloop.counter}}">
            Address: {{ forloop.counter }} </label>
        </div>
      {% endfor %}
        <div class="text-end">
          <button type="submit" class="btn btn-warning mt-3 px-5 fw-bold">
          Continue</button>
        </div>
      </form>
    </div>
  </div>
</div>
{% endblock main-content %}
```

customerregistration.html



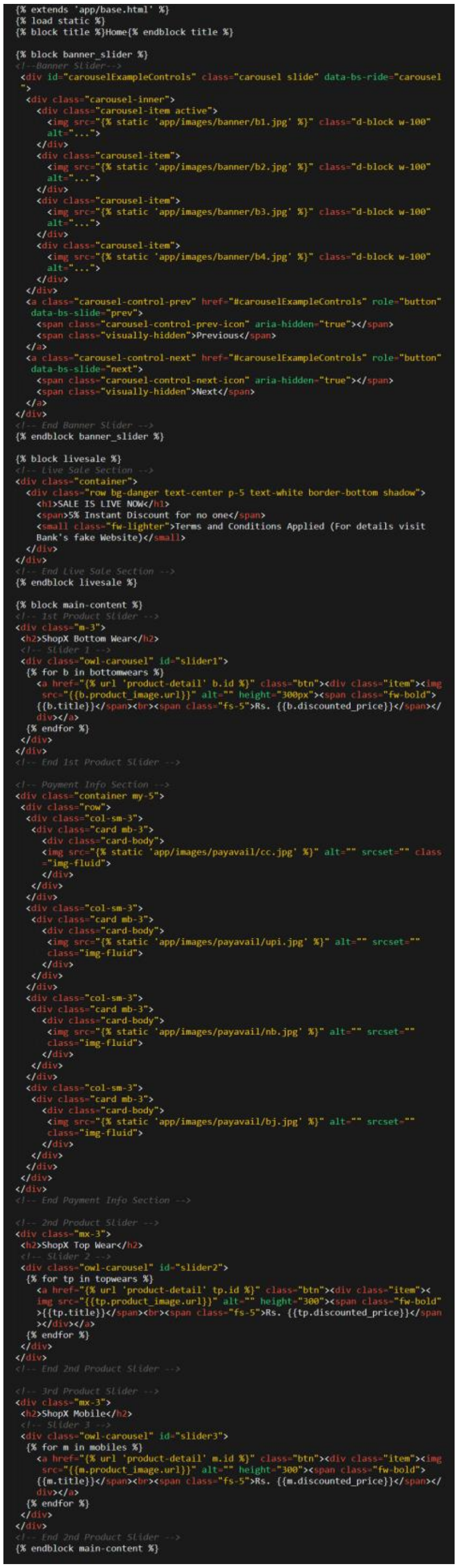
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Customer Registration{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
   <h3>Customer Registration</h3>
   <hr>

   <form action="" method="post" novalidate class="shadow p-5">
    {% if messages %}
    {% for message in messages %}
     <p {% if message.tags %} class="alert alert-{{message.tags}} mb-5" {% endif
      %}>{{message}}</p>
    {% endfor %}
    {% endif %}
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group mb-3">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}
      </small>
      </div>
     {% endfor %}
     <input type="submit" value="Submit" class="btn btn-primary">
     <br>
     <div class="text-center text-primary fw-bold"><small>Existing User ? <a
     href="{% url 'login' %}" class="text-danger">Login Now</a> </small></div>
     {% if form.non_field_errors %}
      {% for error in form.non_field_errors %}
       <p class="alert alert-danger my-3">{{error}}</p>
      {% endfor %}
     {% endif %}
    </form>
  </div>
 </div>
</div>
{% endblock main-content %}
```

emptycart.html



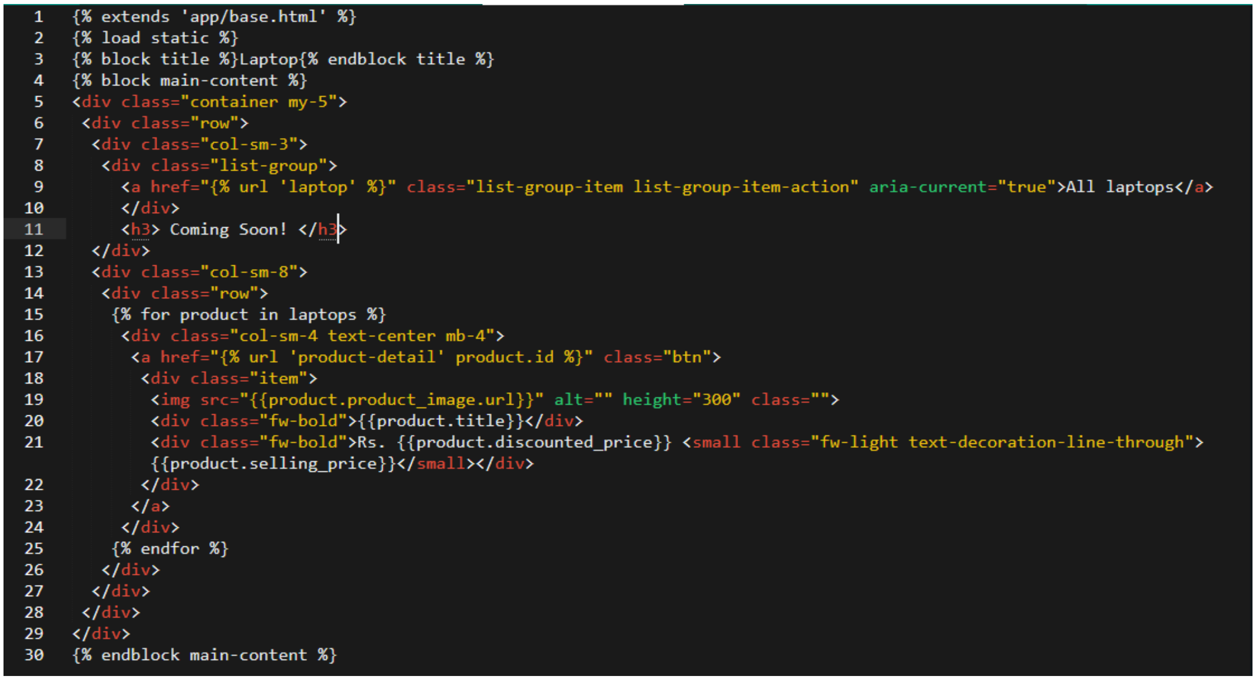
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Empty Cart{% endblock title %}
{% block main-content %}
 <div class="container my-5">
  <div class="row">
   <h3 class="text-center">Ayup! Seems like your cart is empty. Try filling it maybe! :D</h3>
   <div class="text-center">
    <img src="{% static 'app/images/emptycart.png' %}" alt="" class="img-fluid img-thumbnail w-25">
   </div>
  </div>
 </div>
{% endblock main-content %}
```

home.html



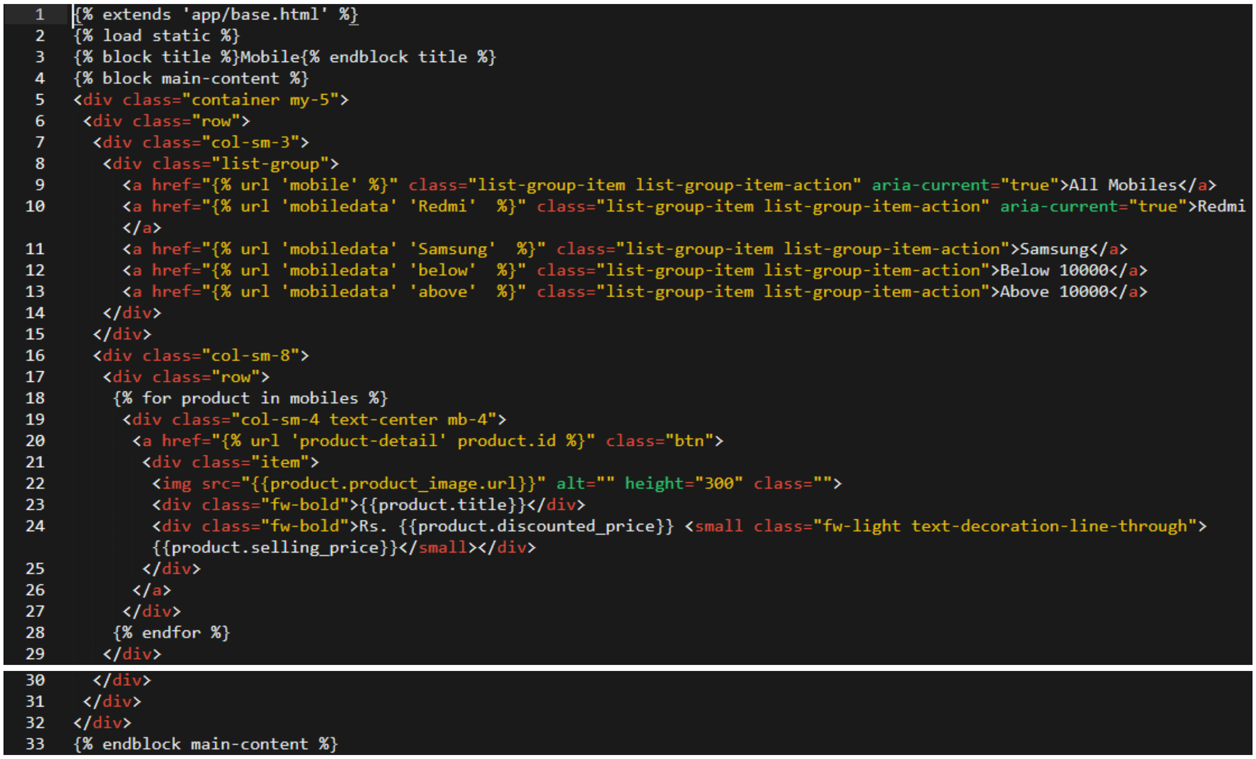
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Home{% endblock title %}

{% block banner_slider %}
<!--Banner Slider-->
 <div id="carouselExampleControls" class="carousel slide" data-bs-ride="carousel
 ">
  <div class="carousel-inner">
    <div class="carousel-item active">
      <img src="{% static 'app/images/banner/b1.jpg' %}" class="d-block w-100"
      alt="...">
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b2.jpg' %}" class="d-block w-100"
      alt="...">
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b3.jpg' %}" class="d-block w-100"
      alt="...">
    </div>
    <div class="carousel-item">
      <img src="{% static 'app/images/banner/b4.jpg' %}" class="d-block w-100"
      alt="...">
    </div>
  </div>
  <a class="carousel-control-prev" href="#carouselExampleControls" role="button"
   data-bs-slide="prev">
    <span class="carousel-control-prev-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Previous</span>
  </a>
  <a class="carousel-control-next" href="#carouselExampleControls" role="button"
   data-bs-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="visually-hidden">Next</span>
  </a>
</div>
<!-- End Banner Slider -->
{% endblock banner_slider %}

{% block livesale %}
<!-- Live Sale Section -->
<div class="container">
  <div class="row bg-danger text-center p-5 text-white border-bottom shadow">
    <h1>SALE IS LIVE NOW</h1>
    <span>5% Instant Discount for no one</span>
    <small class="fw-lighter">Terms and Conditions Applied (For details visit
    Bank's fake Website)</small>
  </div>
</div>
<!-- End Live Sale Section -->
{% endblock livesale %}

{% block main-content %}
<!-- 1st Product Slider -->
<div class="m-3">
 <h2>ShopX Bottom Wear</h2>
 <!-- Slider 1 -->
 <div class="owl-carousel" id="slider1">
  {% for b in bottomwears %}
    <a href="{% url 'product-detail' b.id %}" class="btn"><div class="item"><img
     src="{{b.product_image.url}}" alt="" height="300px"><span class="fw-bold">
    {{b.title}}</span><br><span class="fs-5">Rs. {{b.discounted_price}}</span></
    div></a>
  {% endfor %}
 </div>
</div>
<!-- End 1st Product Slider -->

<!-- Payment Info Section -->
<div class="container my-5">
 <div class="row">
  <div class="col-sm-3">
   <div class="card mb-3">
     <div class="card-body">
     <img src="{% static 'app/images/payavail/cc.jpg' %}" alt="" srcset="" class
     ="img-fluid">
     </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
     <div class="card-body">
      <img src="{% static 'app/images/payavail/upi.jpg' %}" alt="" srcset=""
      class="img-fluid">
     </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
     <div class="card-body">
      <img src="{% static 'app/images/payavail/nb.jpg' %}" alt="" srcset=""
      class="img-fluid">
     </div>
   </div>
  </div>
  <div class="col-sm-3">
   <div class="card mb-3">
     <div class="card-body">
      <img src="{% static 'app/images/payavail/bj.jpg' %}" alt="" srcset=""
      class="img-fluid">
     </div>
   </div>
  </div>
 </div>
</div>
<!-- End Payment Info Section -->

<!-- 2nd Product Slider -->
<div class="mx-3">
 <h2>ShopX Top Wear</h2>
 <!-- Slider 2 -->
 <div class="owl-carousel" id="slider2">
  {% for tp in topwears %}
    <a href="{% url 'product-detail' tp.id %}" class="btn"><div class="item"><
    img src="{{tp.product_image.url}}" alt="" height="300"><span class="fw-bold"
    >{{tp.title}}</span><br><span class="fs-5">Rs. {{tp.discounted_price}}</span
    ></div></a>
  {% endfor %}
 </div>
</div>
<!-- End 2nd Product Slider -->

<!-- 3rd Product Slider -->
<div class="mx-3">
 <h2>ShopX Mobile</h2>
 <!-- Slider 3 -->
 <div class="owl-carousel" id="slider3">
  {% for m in mobiles %}
    <a href="{% url 'product-detail' m.id %}" class="btn"><div class="item"><img
     src="{{m.product_image.url}}" alt="" height="300"><span class="fw-bold">
    {{m.title}}</span><br><span class="fs-5">Rs. {{m.discounted_price}}</span></
    div></a>
  {% endfor %}
 </div>
</div>
<!-- End 2nd Product Slider -->
{% endblock main-content %}
```

laptop.html



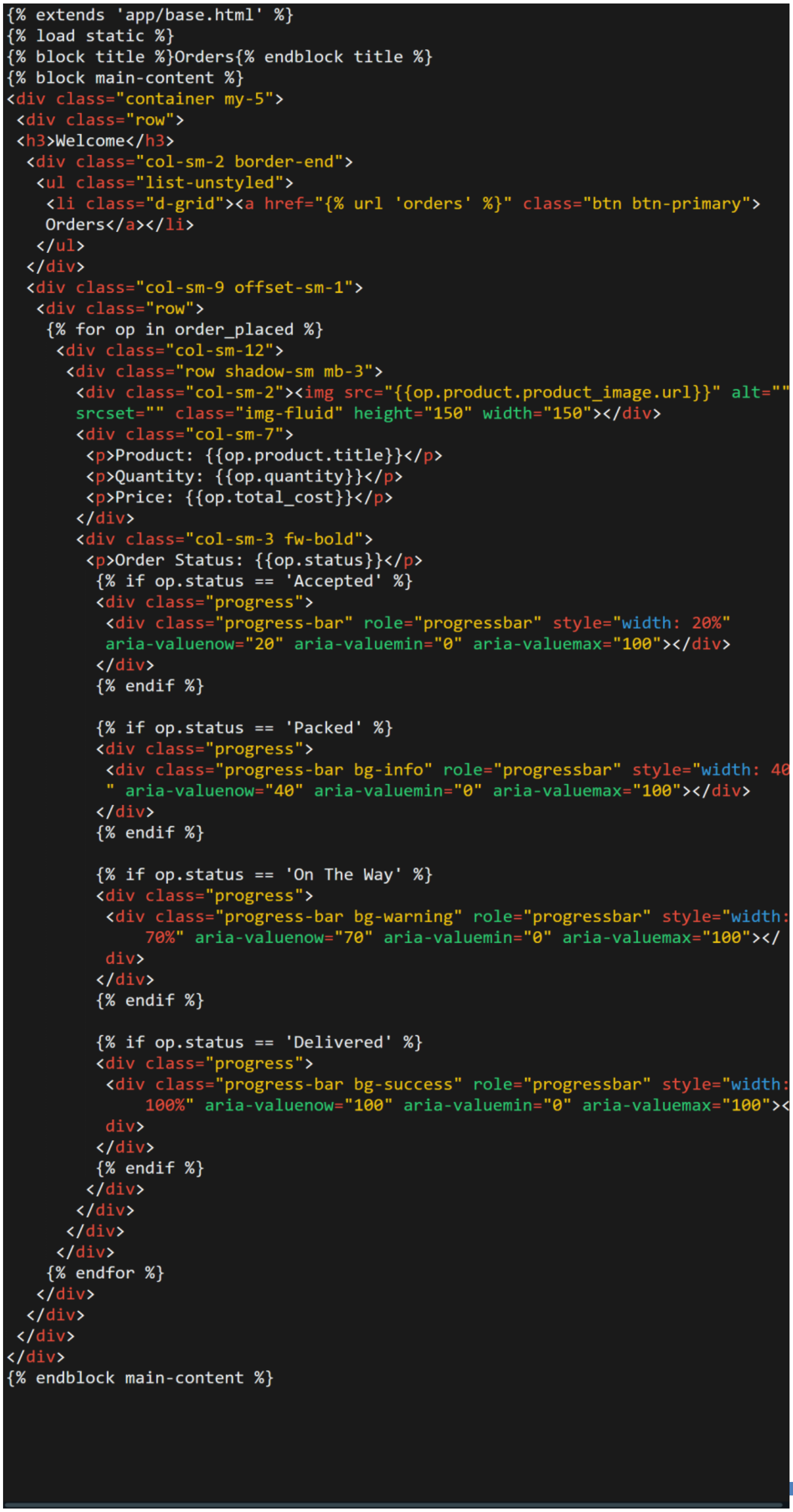
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Laptop{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
  <div class="col-sm-3">
   <div class="list-group">
     <a href="{% url 'laptop' %}" class="list-group-item list-group-item-action" aria-current="true">All laptops</a>
     </div>
     <h3> Coming Soon! </h3>
  </div>
  <div class="col-sm-8">
   <div class="row">
    {% for product in laptops %}
     <div class="col-sm-4 text-center mb-4">
      <a href="{% url 'product-detail' product.id %}" class="btn">
       <div class="item">
        <img src="{{product.product_image.url}}" alt="" height="300" class="">
        <div class="fw-bold">{{product.title}}</div>
        <div class="fw-bold">Rs. {{product.discounted_price}} <small class="fw-light text-decoration-line-through">{{product.selling_price}}</small></div>
       </div>
      </a>
     </div>
    {% endfor %}
   </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

mobile.html



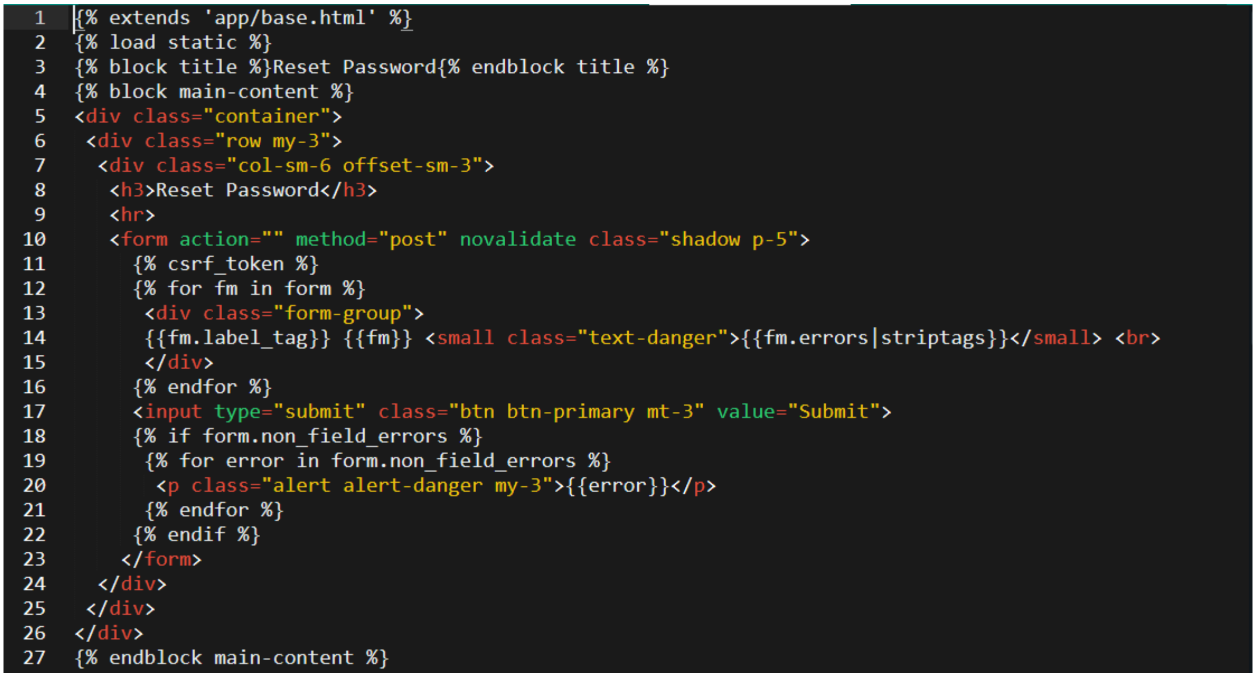
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Mobile{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
  <div class="col-sm-3">
   <div class="list-group">
     <a href="{% url 'mobile' %}" class="list-group-item list-group-item-action" aria-current="true">All Mobiles</a>
     <a href="{% url 'mobiledata' 'Redmi'  %}" class="list-group-item list-group-item-action" aria-current="true">Redmi</a>
     <a href="{% url 'mobiledata' 'Samsung'  %}" class="list-group-item list-group-item-action">Samsung</a>
     <a href="{% url 'mobiledata' 'below'  %}" class="list-group-item list-group-item-action">Below 10000</a>
     <a href="{% url 'mobiledata' 'above'  %}" class="list-group-item list-group-item-action">Above 10000</a>
   </div>
  </div>
  <div class="col-sm-8">
   <div class="row">
    {% for product in mobiles %}
     <div class="col-sm-4 text-center mb-4">
      <a href="{% url 'product-detail' product.id %}" class="btn">
       <div class="item">
        <img src="{{product.product_image.url}}" alt="" height="300" class="">
        <div class="fw-bold">{{product.title}}</div>
        <div class="fw-bold">Rs. {{product.discounted_price}} <small class="fw-light text-decoration-line-through">{{product.selling_price}}</small></div>
       </div>
      </a>
     </div>
    {% endfor %}
   </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

orders.html



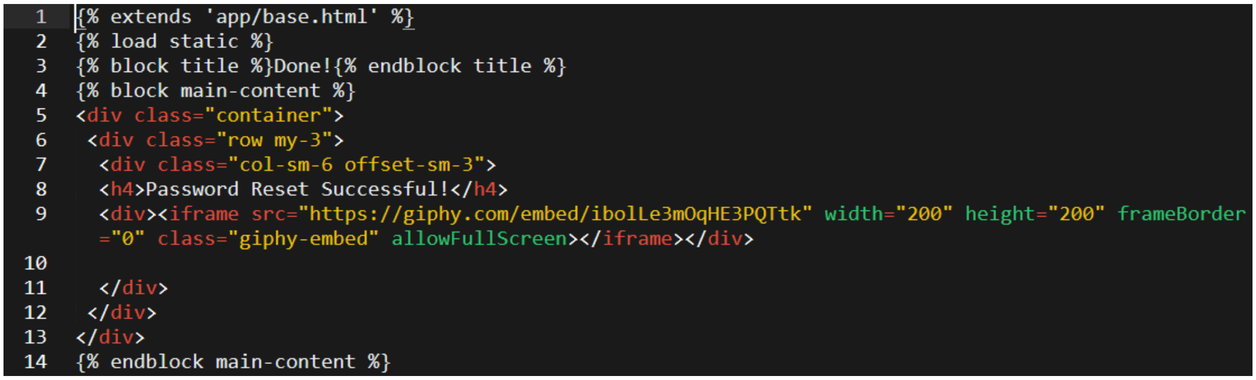
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Orders{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
 <h3>Welcome</h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'orders' %}" class="btn btn-primary">
    Orders</a></li>
   </ul>
  </div>
  <div class="col-sm-9 offset-sm-1">
   <div class="row">
    {% for op in order_placed %}
     <div class="col-sm-12">
      <div class="row shadow-sm mb-3">
       <div class="col-sm-2"><img src="{{op.product.product_image.url}}" alt=""
       srcset="" class="img-fluid" height="150" width="150"></div>
       <div class="col-sm-7">
        <p>Product: {{op.product.title}}</p>
        <p>Quantity: {{op.quantity}}</p>
        <p>Price: {{op.total_cost}}</p>
       </div>
       <div class="col-sm-3 fw-bold">
        <p>Order Status: {{op.status}}</p>
         {% if op.status == 'Accepted' %}
         <div class="progress">
          <div class="progress-bar" role="progressbar" style="width: 20%"
          aria-valuenow="20" aria-valuemin="0" aria-valuemax="100"></div>
         </div>
         {% endif %}

         {% if op.status == 'Packed' %}
         <div class="progress">
          <div class="progress-bar bg-info" role="progressbar" style="width: 40
          " aria-valuenow="40" aria-valuemin="0" aria-valuemax="100"></div>
         </div>
         {% endif %}

         {% if op.status == 'On The Way' %}
         <div class="progress">
          <div class="progress-bar bg-warning" role="progressbar" style="width:
              70%" aria-valuenow="70" aria-valuemin="0" aria-valuemax="100"></
          div>
         </div>
         {% endif %}

         {% if op.status == 'Delivered' %}
         <div class="progress">
          <div class="progress-bar bg-success" role="progressbar" style="width:
              100%" aria-valuenow="100" aria-valuemin="0" aria-valuemax="100"><
          div>
         </div>
         {% endif %}
        </div>
       </div>
      </div>
     </div>
    {% endfor %}
   </div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

password_reset.html



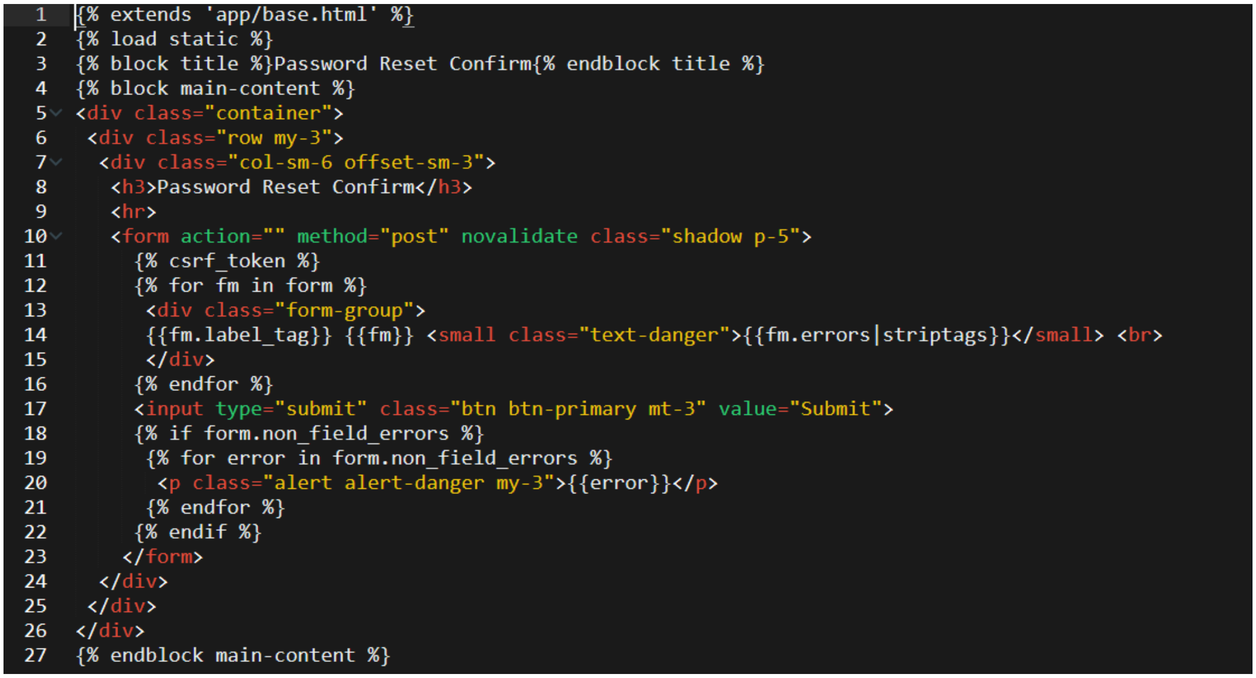
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Reset Password{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
   <h3>Reset Password</h3>
   <hr>
   <form action="" method="post" novalidate class="shadow p-5">
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}</small> <br>
      </div>
     {% endfor %}
     <input type="submit" class="btn btn-primary mt-3" value="Submit">
     {% if form.non_field_errors %}
      {% for error in form.non_field_errors %}
       <p class="alert alert-danger my-3">{{error}}</p>
      {% endfor %}
     {% endif %}
    </form>
  </div>
 </div>
</div>
{% endblock main-content %}
```

password_reset_complete.html



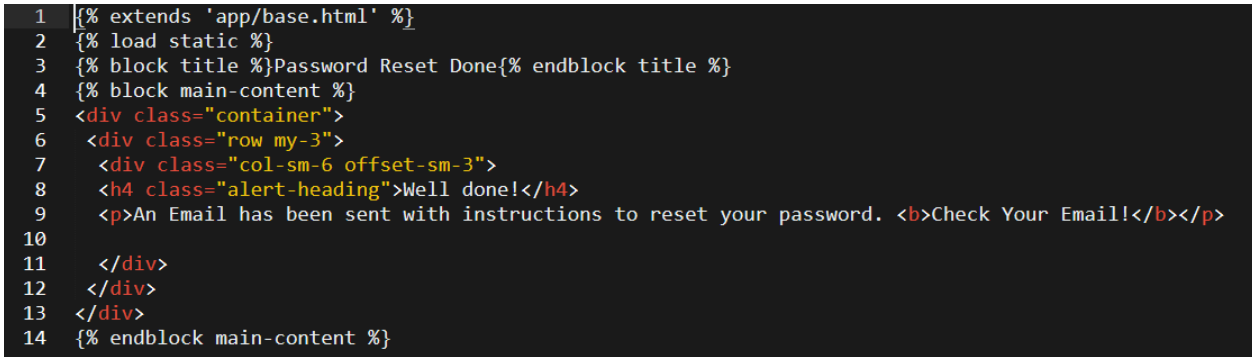
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Done!{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
  <h4>Password Reset Successful!</h4>
  <div><iframe src="https://giphy.com/embed/ibolLe3mOqHE3PQTtk" width="200" height="200" frameBorder="0" class="giphy-embed" allowFullScreen></iframe></div>

  </div>
 </div>
</div>
{% endblock main-content %}
```

password_reset_confirm.html



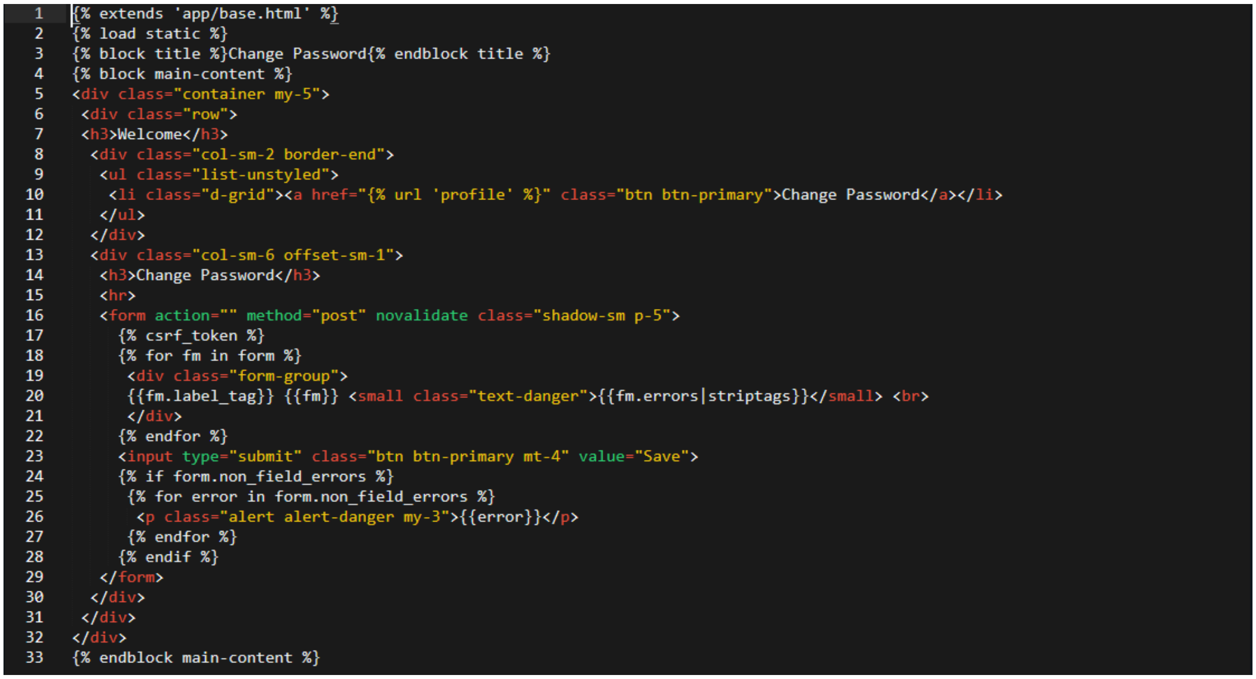
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Password Reset Confirm{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
   <h3>Password Reset Confirm</h3>
   <hr>
   <form action="" method="post" novalidate class="shadow p-5">
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}</small> <br>
      </div>
     {% endfor %}
     <input type="submit" class="btn btn-primary mt-3" value="Submit">
     {% if form.non_field_errors %}
      {% for error in form.non_field_errors %}
       <p class="alert alert-danger my-3">{{error}}</p>
      {% endfor %}
     {% endif %}
    </form>
  </div>
 </div>
</div>
{% endblock main-content %}
```

password_reset_done.html



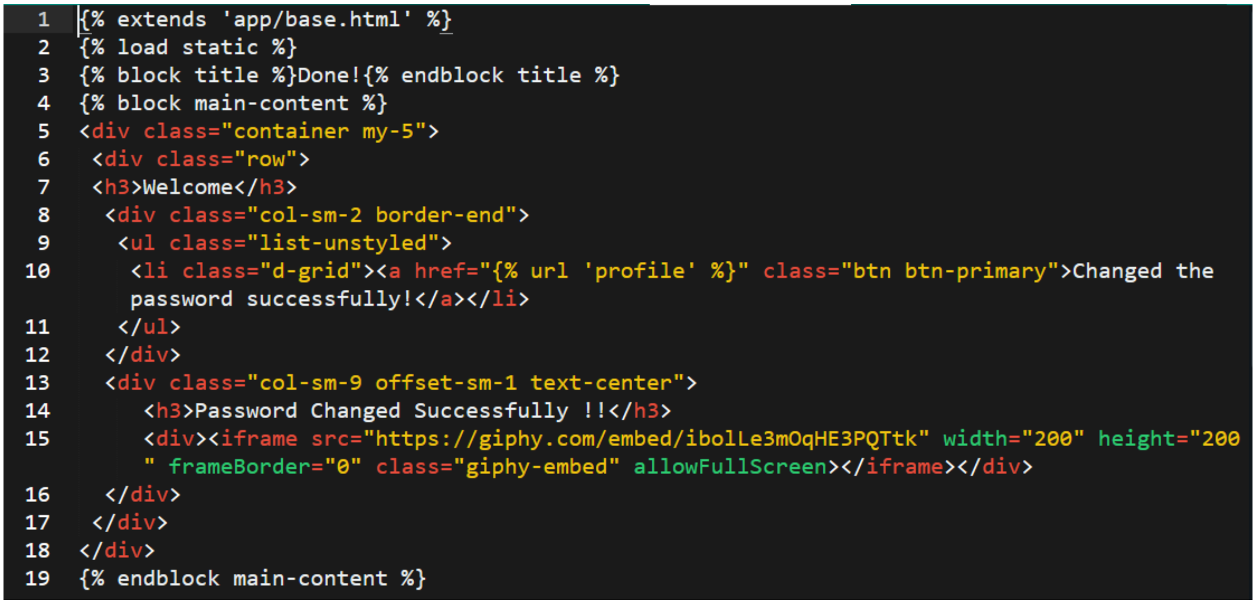
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Password Reset Done{% endblock title %}
{% block main-content %}
<div class="container">
 <div class="row my-3">
  <div class="col-sm-6 offset-sm-3">
  <h4 class="alert-heading">Well done!</h4>
  <p>An Email has been sent with instructions to reset your password. <b>Check Your Email!</b></p>

  </div>
 </div>
</div>
{% endblock main-content %}
```

passwordchange.html



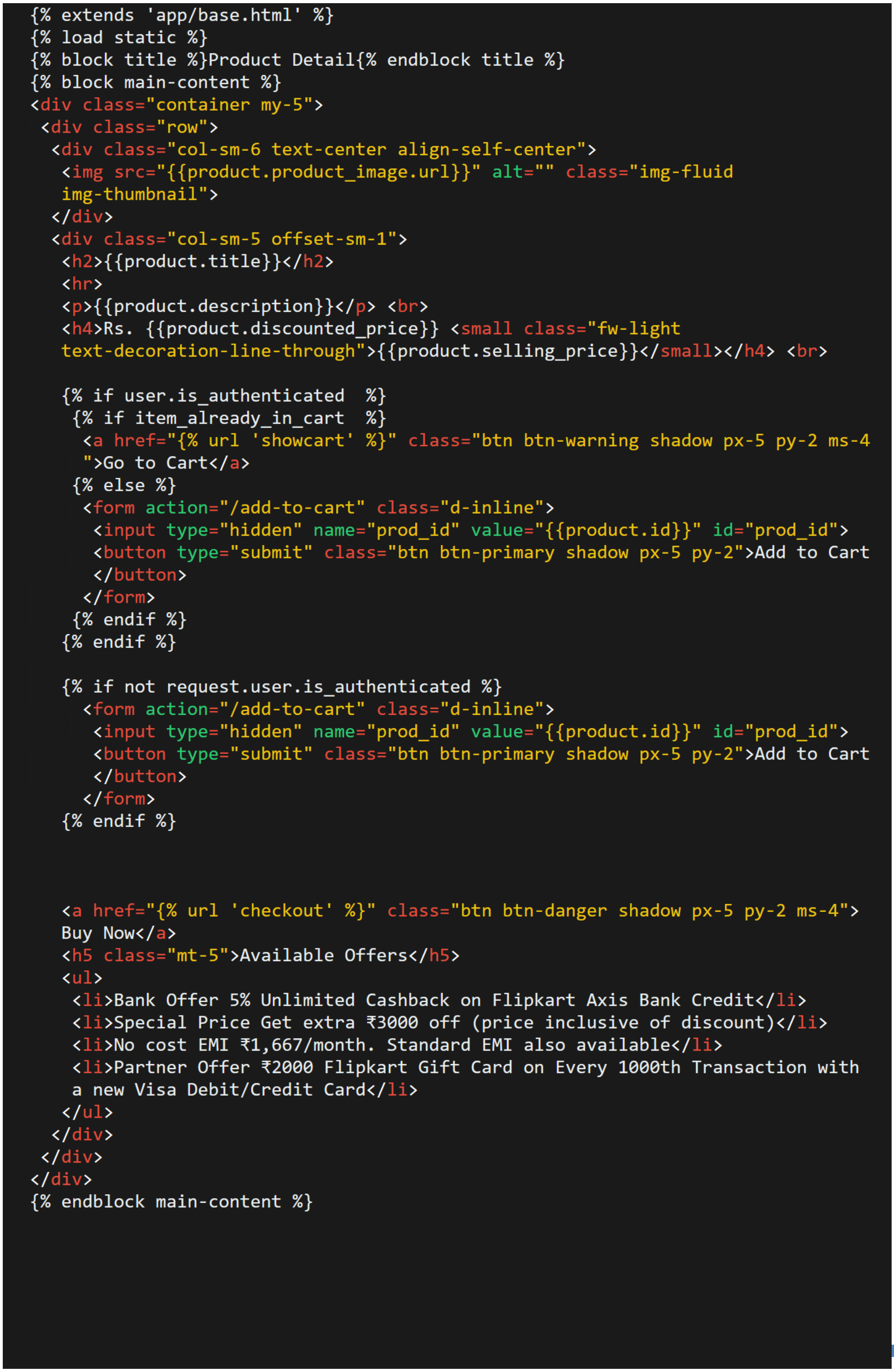
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Change Password{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
 <h3>Welcome</h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'profile' %}" class="btn btn-primary">Change Password</a></li>
   </ul>
  </div>
  <div class="col-sm-6 offset-sm-1">
   <h3>Change Password</h3>
   <hr>
   <form action="" method="post" novalidate class="shadow-sm p-5">
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}</small> <br>
      </div>
     {% endfor %}
     <input type="submit" class="btn btn-primary mt-4" value="Save">
     {% if form.non_field_errors %}
      {% for error in form.non_field_errors %}
       <p class="alert alert-danger my-3">{{error}}</p>
      {% endfor %}
     {% endif %}
   </form>
  </div>
 </div>
</div>
{% endblock main-content %}
```

passwordchangedone.html



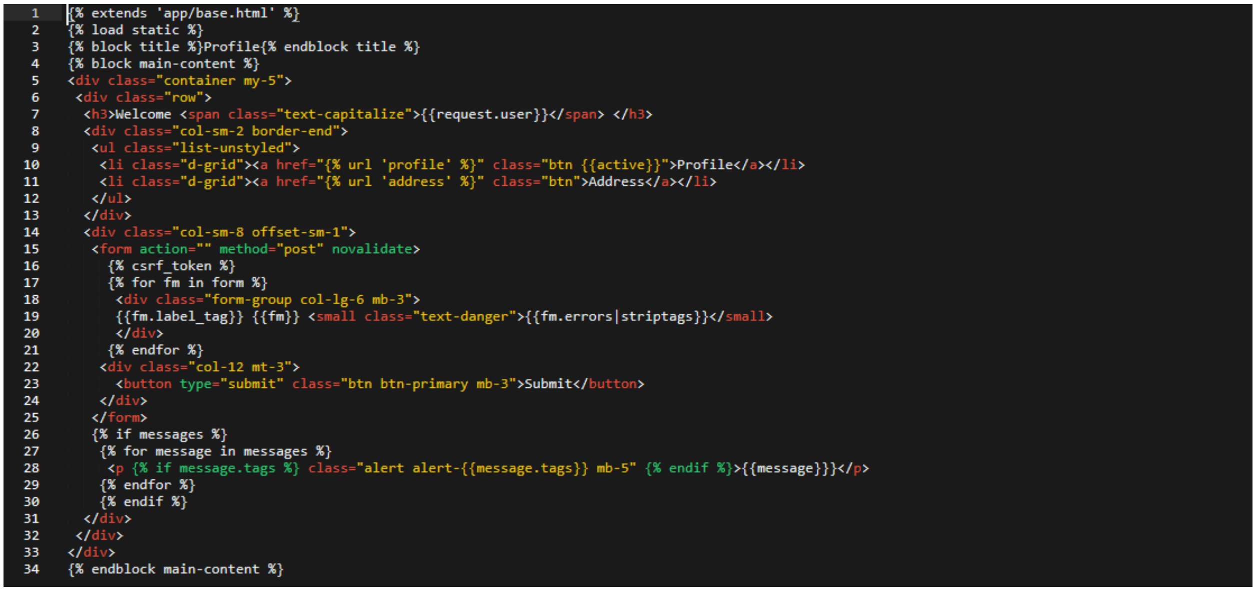
```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Done!{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
 <h3>Welcome</h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'profile' %}" class="btn btn-primary">Changed the password successfully!</a></li>
   </ul>
  </div>
  <div class="col-sm-9 offset-sm-1 text-center">
     <h3>Password Changed Successfully !!</h3>
     <div><iframe src="https://giphy.com/embed/ibolLe3mOqHE3PQTtk" width="200" height="200" frameBorder="0" class="giphy-embed" allowFullScreen></iframe></div>
  </div>
 </div>
</div>
{% endblock main-content %}
```

productdetail.html



```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Product Detail{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
  <div class="col-sm-6 text-center align-self-center">
   <img src="{{product.product_image.url}}" alt="" class="img-fluid
   img-thumbnail">
  </div>
  <div class="col-sm-5 offset-sm-1">
   <h2>{{product.title}}</h2>
   <hr>
   <p>{{product.description}}</p> <br>
   <h4>Rs. {{product.discounted_price}} <small class="fw-light
   text-decoration-line-through">{{product.selling_price}}</small></h4> <br>

   {% if user.is_authenticated  %}
    {% if item_already_in_cart  %}
     <a href="{% url 'showcart' %}" class="btn btn-warning shadow px-5 py-2 ms-4
     ">Go to Cart</a>
    {% else %}
     <form action="/add-to-cart" class="d-inline">
      <input type="hidden" name="prod_id" value="{{product.id}}" id="prod_id">
      <button type="submit" class="btn btn-primary shadow px-5 py-2">Add to Cart
      </button>
     </form>
    {% endif %}
   {% endif %}

   {% if not request.user.is_authenticated %}
     <form action="/add-to-cart" class="d-inline">
      <input type="hidden" name="prod_id" value="{{product.id}}" id="prod_id">
      <button type="submit" class="btn btn-primary shadow px-5 py-2">Add to Cart
      </button>
     </form>
   {% endif %}



   <a href="{% url 'checkout' %}" class="btn btn-danger shadow px-5 py-2 ms-4">
   Buy Now</a>
   <h5 class="mt-5">Available Offers</h5>
   <ul>
    <li>Bank Offer 5% Unlimited Cashback on Flipkart Axis Bank Credit</li>
    <li>Special Price Get extra ₹3000 off (price inclusive of discount)</li>
    <li>No cost EMI ₹1,667/month. Standard EMI also available</li>
    <li>Partner Offer ₹2000 Flipkart Gift Card on Every 1000th Transaction with
    a new Visa Debit/Credit Card</li>
   </ul>
  </div>
 </div>
</div>
{% endblock main-content %}
```

profile.html



```
{% extends 'app/base.html' %}
{% load static %}
{% block title %}Profile{% endblock title %}
{% block main-content %}
<div class="container my-5">
 <div class="row">
  <h3>Welcome <span class="text-capitalize">{{request.user}}</span> </h3>
  <div class="col-sm-2 border-end">
   <ul class="list-unstyled">
    <li class="d-grid"><a href="{% url 'profile' %}" class="btn {{active}}">Profile</a></li>
    <li class="d-grid"><a href="{% url 'address' %}" class="btn">Address</a></li>
   </ul>
  </div>
  <div class="col-sm-8 offset-sm-1">
   <form action="" method="post" novalidate>
     {% csrf_token %}
     {% for fm in form %}
      <div class="form-group col-lg-6 mb-3">
      {{fm.label_tag}} {{fm}} <small class="text-danger">{{fm.errors|striptags}}</small>
      </div>
     {% endfor %}
    <div class="col-12 mt-3">
      <button type="submit" class="btn btn-primary mb-3">Submit</button>
    </div>
   </form>
   {% if messages %}
    {% for message in messages %}
     <p {% if message.tags %} class="alert alert-{{message.tags}} mb-5" {% endif %}>{{message}}</p>
    {% endfor %}
    {% endif %}
  </div>
 </div>
</div>
{% endblock main-content %}
```

Step 5 | Adding CSS, JS and image files

Now that you've created the necessary front-end websites, it's time to import CSS, JS and image files

I highly recommend you to get all of them from my GitHub.

Step 6 | Running the project

sigh Finally! The time has come, to execute our project. I hope you've followed the steps I've mentioned here and now you are ready to roll!

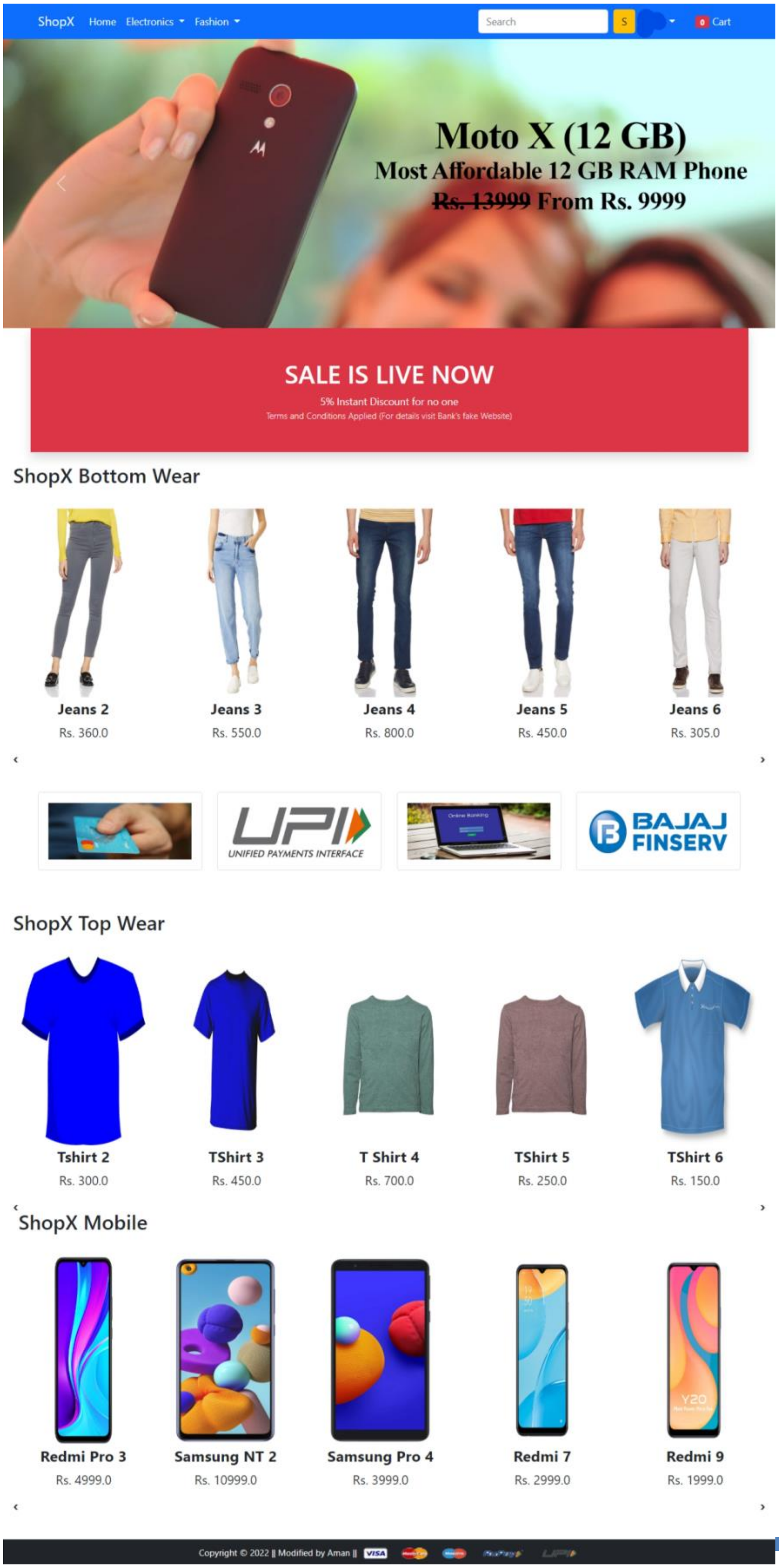
So, open a cmd window at your manage.py location and type **py manage.py migrate**

Hit enter and then type **py manage.py createsuperuser**

This will open a CLI Interface to create an admin account. Enter your details and then, type **py manage.py runserver**.

BOOM!

You should see a website like this



ShopX Home Electronics ▾ Fashion ▾ Search S ▾ 0 Cart

Moto X (12 GB)

Most Affordable 12 GB RAM Phone

~~Rs. 13999~~ From Rs. 9999

SALE IS LIVE NOW

5% Instant Discount for no one

Terms and Conditions Applied (For details visit Bank's fake Website)

ShopX Bottom Wear

Jeans 2
Rs. 360.0

Jeans 3
Rs. 550.0

Jeans 4
Rs. 800.0

Jeans 5
Rs. 450.0

Jeans 6
Rs. 305.0

‹ ›

ShopX Top Wear

Tshirt 2
Rs. 300.0

TShirt 3
Rs. 450.0

T Shirt 4
Rs. 700.0

TShirt 5
Rs. 250.0

TShirt 6
Rs. 150.0

‹ ›

ShopX Mobile

Redmi Pro 3
Rs. 4999.0

Samsung NT 2
Rs. 10999.0

Samsung Pro 4
Rs. 3999.0

Redmi 7
Rs. 2999.0

Redmi 9
Rs. 1999.0

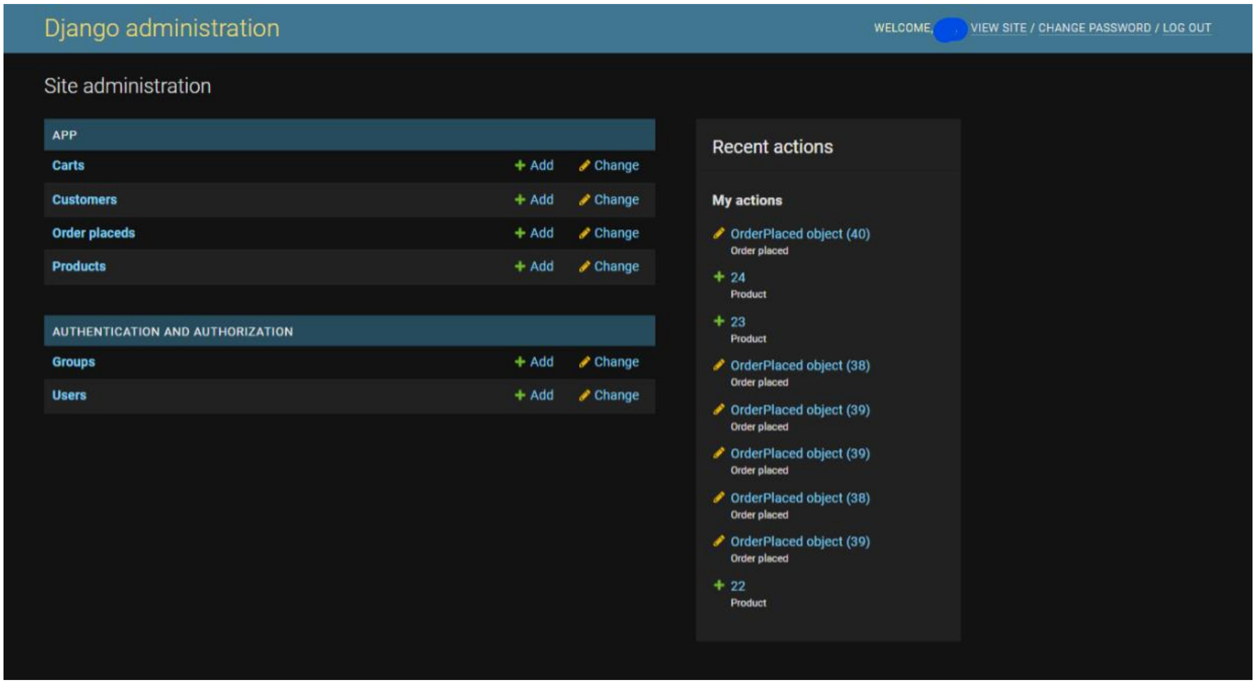
‹ ›

Copyright © 2022 || Modified by Aman ||

Step 7 | Accessing the admin panel

Now as you have all the stuff loaded up and running, it's time to finally add some items to your website, right? So, head over to localhost:8000/admin.

This will take you to an interface similar to this



Django administration

WELCOME, [redacted] VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

APP		
Carts	+ Add	Change
Customers	+ Add	Change
Order placeds	+ Add	Change
Products	+ Add	Change

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change

Recent actions

My actions

- OrderPlaced object (40)
  Order placed
- + 24
  Product
- + 23
  Product
- OrderPlaced object (38)
  Order placed
- OrderPlaced object (39)
  Order placed
- OrderPlaced object (39)
  Order placed
- OrderPlaced object (38)
  Order placed
- OrderPlaced object (39)
  Order placed
- + 22
  Product

Here you can click **Products** and it will take you to a product addition wizard! Pretty neat huh!

Troubleshooting

Any database errors?

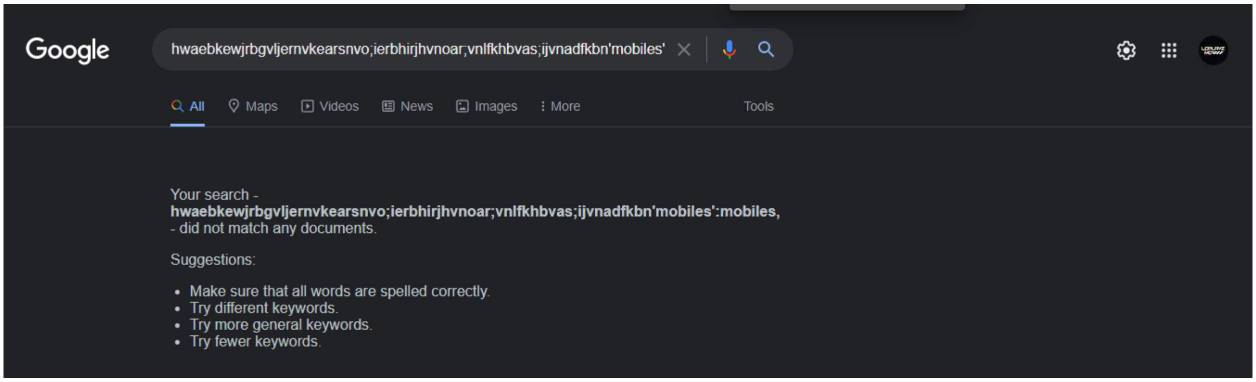
- ➢ I recommend you to take the **db.sqlite3** file from my GitHub, and try replacing it with the original one if that works. It will also provide you with some default products.

Note

Be sure to grab the **media** folder from my GitHub as it contains all the image files and paste it into the main directory.

Bonus | Adding 404 Error Page

Every website isn’t perfect nor complete. Let’s take an example. On Google, search some random gibberish like,


Google
hwaebkewjrbgvljernvkearsnvo;ierbhirjhvnoar;vnlfkhbvas;ijvnadfkbn'mobiles'
All
Maps
Videos
News
Images
More
Tools

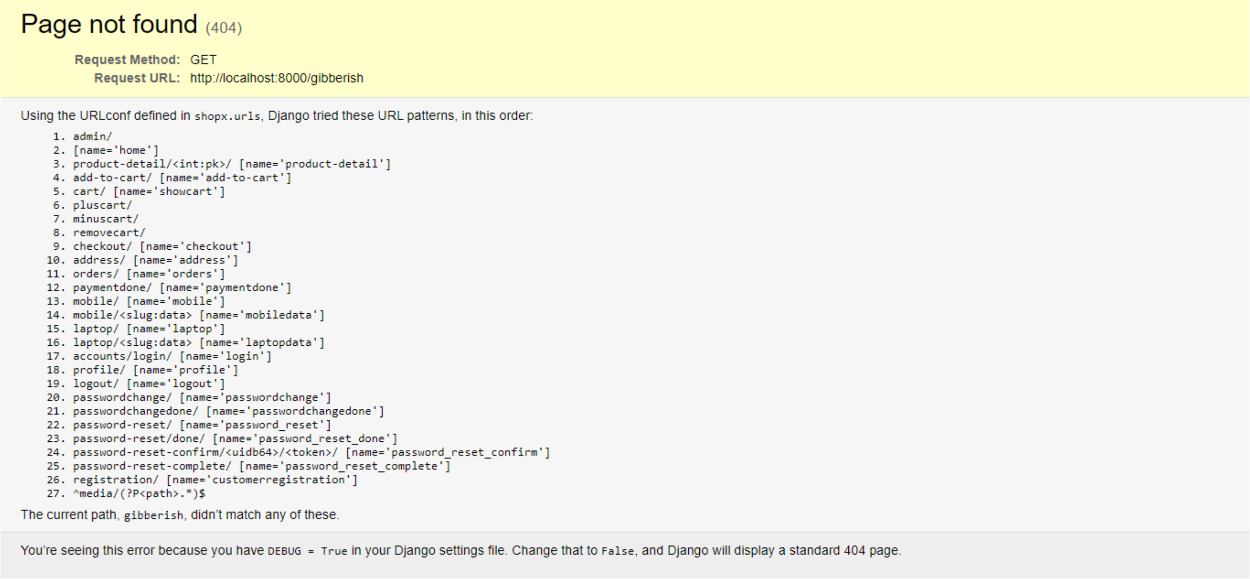
Your search - **hwaebkewjrbgvljernvkearsnvo;ierbhirjhvnoar;vnlfkhbvas;ijvnadfkbn'mobiles':mobiles,** - did not match any documents.

Suggestions:

- Make sure that all words are spelled correctly.
- Try different keywords.
- Try more general keywords.
- Try fewer keywords.

Yea? See that page? Here Google returned with *Your search did not match any documents.*

You don’t want customers to go anywhere and see this,



Page not found (404)

Request Method: GET
Request URL: http://localhost:8000/gibberish

Using the URLconf defined in `shopx.urls`, Django tried these URL patterns, in this order:

```
 1. admin/
 2. [name='home']
 3. product-detail/<int:pk>/ [name='product-detail']
 4. add-to-cart/ [name='add-to-cart']
 5. cart/ [name='showcart']
 6. pluscart/
 7. minuscart/
 8. removecart/
 9. checkout/ [name='checkout']
10. address/ [name='address']
11. orders/ [name='orders']
12. paymentdone/ [name='paymentdone']
13. mobile/ [name='mobile']
14. mobile/<slug:data> [name='mobiledata']
15. laptop/ [name='laptop']
16. laptop/<slug:data> [name='laptopdata']
17. accounts/login/ [name='login']
18. profile/ [name='profile']
19. logout/ [name='logout']
20. passwordchange/ [name='passwordchange']
21. passwordchangedone/ [name='passwordchangedone']
22. password-reset/ [name='password_reset']
23. password-reset/done/ [name='password_reset_done']
24. password-reset-confirm/<uidb64>/<token>/ [name='password_reset_confirm']
25. password-reset-complete/ [name='password_reset_complete']
26. registration/ [name='customerregistration']
27. ^media/(?P<path>.*)$
```

The current path, `gibberish`, didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

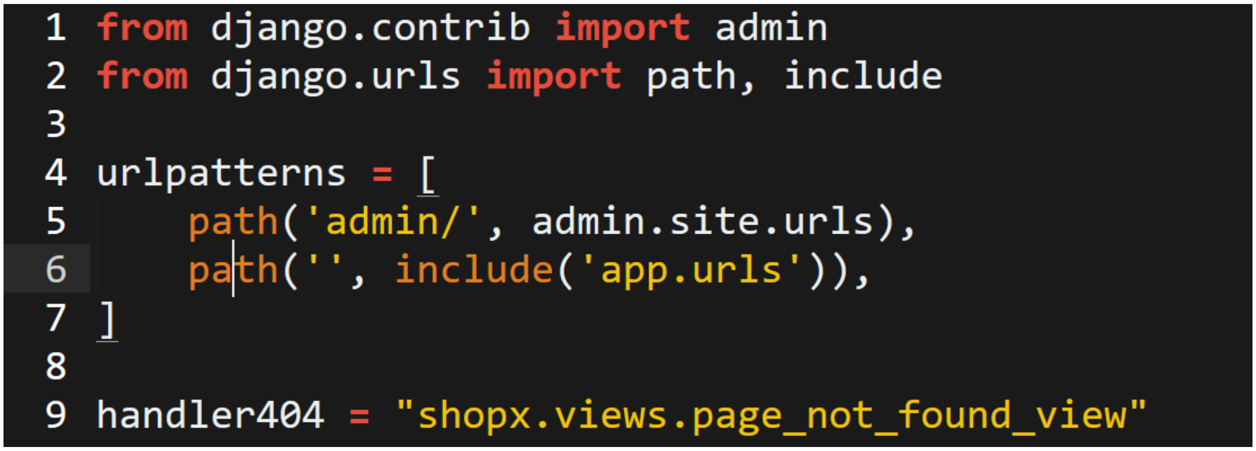
We need to make it a subtle error page with a button to take those lost users to the home page, right?

We are going to implement just that right now.

See when you execute the website right now, it will show an error. That's because we have some stuff related to the 404 page already done! Now it's time to finish it.

Adding 404 Page

You might remember that we have added line 9 at <project>'s urls.py. Reference



```
1 from django.contrib import admin
2 from django.urls import path, include
3
4 urlpatterns = [
5     path('admin/', admin.site.urls),
6     path('', include('app.urls')),
7 ]
8
9 handler404 = "shopx.views.page_not_found_view"
```

This line was for the 404 page. Now the **handler404** you might see is a default **Django** term. And **shopx.views.page_not_found_view** refers a view in <project>'s views.py . But now you might think, why should we add views.py in <project>'s folder? Isn't it supposed that we should add it in-app folder? Well, you are right, but here we want to address every link which is not defined, such as http://localhost:8000/gibberish and all, to redirect to a specific page that will have a button right? So basically, Think it this way…

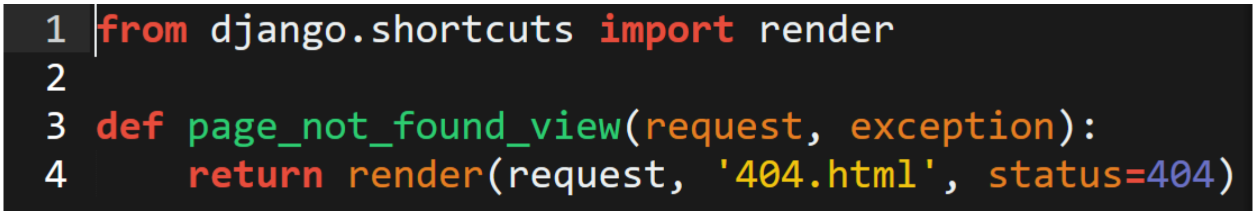
<project> is the *parent* of the **app** folder, so, any invalid links should redirect to the parent so the parent handles it.

So first, let's create a file named **views.py** in the <project> folder

Your directory should look something like this,

```
.
└── <project_name>
    └── <project>
        ├── __pycache__
        ├── __init__.py
        ├── asgi.py
        ├── settings.py
        ├── urls.py
        ├── views.py <- Create this file
        └── wsgi.py
```

Now in this file, add this bit of code,



```python
from django.shortcuts import render

def page_not_found_view(request, exception):
    return render(request, '404.html', status=404)
```

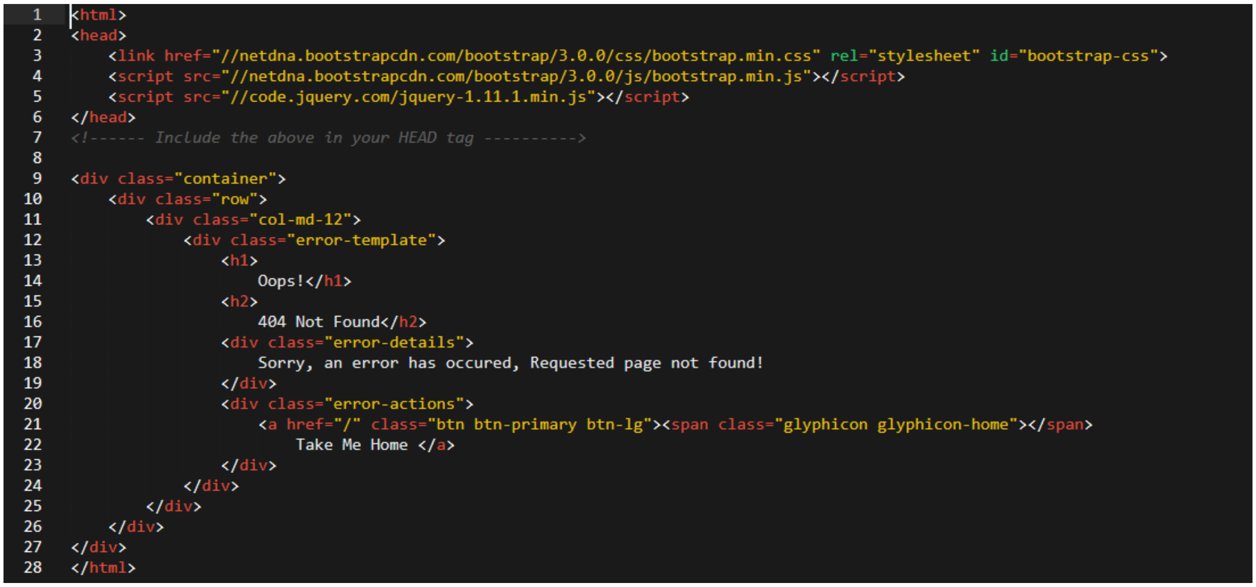
Now you might notice that we have called the 404.html file, now we are going to make that file.

Create a new folder named **templates** and create a new file named **404.html** in the <project_name> folder. Your directory should look like this,

```
.
└── <project_name>
    ├── <project>
    ├── app
    └── templates <- Create this folder
        └── 404.html
```

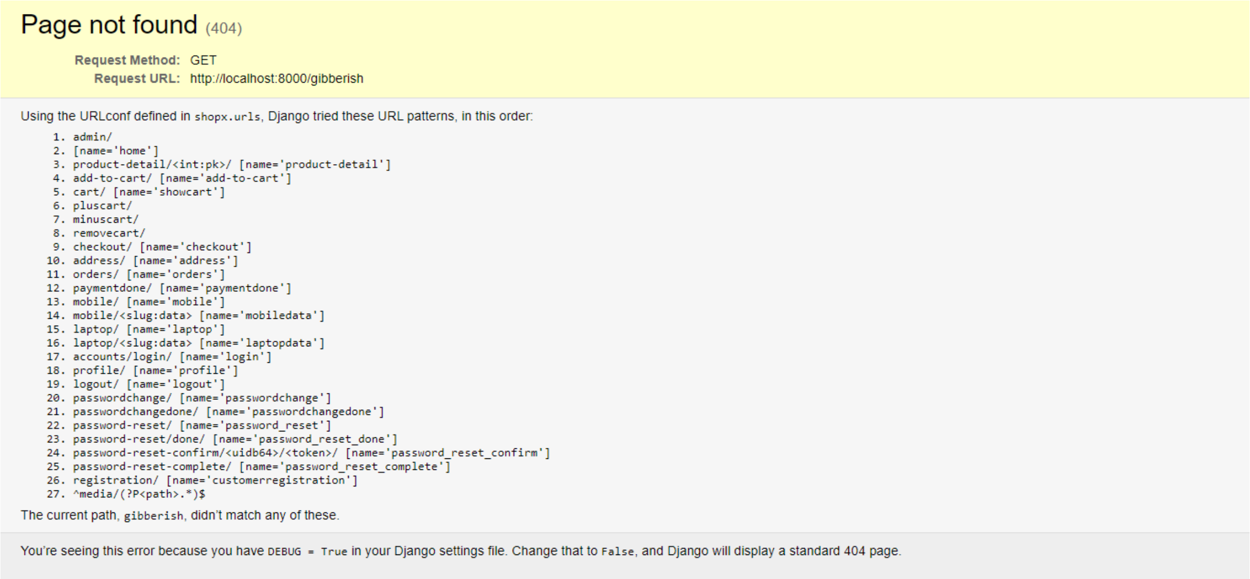
Now in **404.html**, we are going to add this bit of code,

Note that you can add any kind of code here so it isn't necessary to use just my code but, well … here I go anyways :D



```html
<html>
<head>
    <link href="//netdna.bootstrapcdn.com/bootstrap/3.0.0/css/bootstrap.min.css" rel="stylesheet" id="bootstrap-css">
    <script src="//netdna.bootstrapcdn.com/bootstrap/3.0.0/js/bootstrap.min.js"></script>
    <script src="//code.jquery.com/jquery-1.11.1.min.js"></script>
</head>
<!------ Include the above in your HEAD tag ---------->

<div class="container">
    <div class="row">
        <div class="col-md-12">
            <div class="error-template">
                <h1>
                    Oops!</h1>
                <h2>
                    404 Not Found</h2>
                <div class="error-details">
                    Sorry, an error has occured, Requested page not found!
                </div>
                <div class="error-actions">
                    <a href="/" class="btn btn-primary btn-lg"><span class="glyphicon glyphicon-home"></span>
                        Take Me Home </a>
                </div>
            </div>
        </div>
    </div>
</div>
</html>
```

Now that the 404.html is ready, let's also make some changes to **settings.py**. Django itself recommends turning off **DEBUG=TRUE** in settings.py when in production.



Page not found (404)

Request Method: GET
Request URL: http://localhost:8000/gibberish

Using the URLconf defined in `shopx.urls`, Django tried these URL patterns, in this order:

```
 1. admin/
 2. [name='home']
 3. product-detail/<int:pk>/ [name='product-detail']
 4. add-to-cart/ [name='add-to-cart']
 5. cart/ [name='showcart']
 6. pluscart/
 7. minuscart/
 8. removecart/
 9. checkout/ [name='checkout']
10. address/ [name='address']
11. orders/ [name='orders']
12. paymentdone/ [name='paymentdone']
13. mobile/ [name='mobile']
14. mobile/<slug:data> [name='mobiledata']
15. laptop/ [name='laptop']
16. laptop/<slug:data> [name='laptopdata']
17. accounts/login/ [name='login']
18. profile/ [name='profile']
19. logout/ [name='logout']
20. passwordchange/ [name='passwordchange']
21. passwordchangedone/ [name='passwordchangedone']
22. password-reset/ [name='password_reset']
23. password-reset/done/ [name='password_reset_done']
24. password-reset-confirm/<uidb64>/<token>/ [name='password_reset_confirm']
25. password-reset-complete/ [name='password_reset_complete']
26. registration/ [name='customerregistration']
27. ^media/(?P<path>.*)$
```

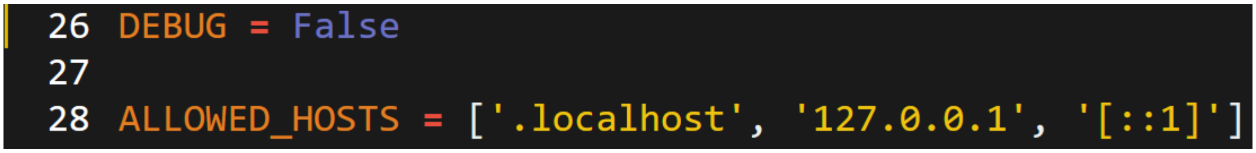
The current path, `gibberish`, didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

Here if you take a look at the footer, you might see:
You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page

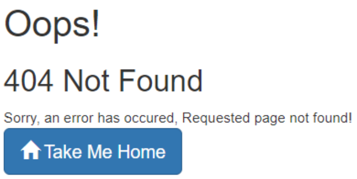
We want to change this,

Head over to **settings.py** and change these lines



```
DEBUG = False

ALLOWED_HOSTS = ['.localhost', '127.0.0.1', '[::1]']
```

Note that you can change **ALLOWED_HOSTS** to your domain, for example, you've set your home router, and in the router settings you've set a domain, now you want your app to use that domain, then you should set it here as a string and voila! Never mind… so then you want to set **DEBUG** = **False**. Now when you visit any invalid link, it will redirect you to this page.



Oops!

404 Not Found

Sorry, an error has occured, Requested page not found!

So that's it! Now your lost customers will find a way home! Hurray!

So that's it for this documentation, I hope you liked it! If you find any errors or doubts, be sure to visit my GitHub's issues page and file an issue there. I will do my best to answer those.

References

GitHub

Django

Project Provider

Author