Initailly you have to run the commands

Python manage.py runserver –to check the server running status

Then

Python manage.py startapp review –to stat the app named review

And also in the cmd you have to run

Python manage.py makemigrations

Python manage.py sqlmigrate review 0001

Python manage.py sqlmigrate review 0002

Python manage.py migrate

Code

In ecommerce folder

Asgi.py

```
"""
ASGI config for ecommerce project.

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

For more information on this file, see
https://docs.djangoproject.com/en/3.1/howto/deployment/asgi/
"""

import os

from django.core.asgi import get_asgi_application

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

application = get_asgi_application()
```

Settings.py

```
import os
"""
Django settings for ecommerce project.

Generated by 'django-admin startproject' using Django 3.1.6.

For more information on this file, see
https://docs.djangoproject.com/en/3.1/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.1/ref/settings/
"""
```

```
from pathlib import Path

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent


# Quick-start development settings - unsuitable for production
# See https://docs.djangoproject.com/en/3.1/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '9+!a(^%io*f!77ipt3i*8tja%14_0$s5mo$vcho77wd76k_kgh'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = []


# Application definition

INSTALLED_APPS = [
    'review.apps.ReviewConfig',
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]

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

ROOT_URLCONF = 'ecommerce.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(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',
            ],
        },
```

```
    },
]

WSGI_APPLICATION = 'ecommerce.wsgi.application'


# Database
# https://docs.djangoproject.com/en/3.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'ecommerce',
        'USER': 'postgres',
        'PASSWORD':'1234',
        'HOST':'localhost'
    }
}


# Password validation
# https://docs.djangoproject.com/en/3.1/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/3.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.1/howto/static-files/
```

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

Urls.py

```
"""ecommerce URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.1/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path,include

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

Wsgi.py

```
"""
WSGI config for ecommerce project.

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

For more information on this file, see
https://docs.djangoproject.com/en/3.1/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

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

application = get_wsgi_application()
```

REVIEW APP

--pycache-

0001_initial.py

```
# Generated by Django 3.1.6 on 2021-02-15 09:10

from django.db import migrations, models

#initially creted table for migration

class Migration(migrations.Migration):

    initial = True

    dependencies = [
    ]

    operations = [
        migrations.CreateModel(
            name='customerReview',
            fields=[
                ('id', models.AutoField(auto_created=True, primary_key=True, serialize=False
, verbose_name='ID')),
                ('user_name', models.CharField(max_length=100)),
                ('bran_name', models.CharField(max_length=100)),
            ],
        ),
    ]
```

002_auto_2021…py

```
# Generated by Django 3.1.6 on 2021-02-15 09:15

from django.db import migrations, models

#adding the additional columns to the database created (initially I forgot to add them so I
altered the table using this other migration)

class Migration(migrations.Migration):

    dependencies = [
        ('review', '0001_initial'),
    ]

    operations = [
        migrations.AddField(
            model_name='customerreview',
            name='produ_name',
            field=models.CharField(default=0, max_length=100),
            preserve_default=False,
        ),
        migrations.AddField(
            model_name='customerreview',
```

```
            name='review_data',
            field=models.TextField(default='nokia'),
            preserve_default=False,
        ),
    ]
```

Admin.py

```
from django.contrib import admin
from .models import customerReview
# Register your models here.
admin.site.register(customerReview) #this is used to create the admin credential
```

apps.py

```
from django.apps import AppConfig


class ReviewConfig(AppConfig):
    name = 'review'           #name of the app configured
```

models.py

```
from django.db import models

# Create your models here.
class customerReview(models.Model):
    user_name=models.CharField(max_length=100) #model for storing user name
    bran_name=models.CharField(max_length=100)  #model for storing brand name
    produ_name=models.CharField(max_length=100)  #model for storing product name
    review_data=models.TextField()
```

urls.py

```
from django.urls import path
from . import views
urlpatterns = [
    path('', views.generate, name='generate'),     #navigates the url to the home page descr
ibing function
    path('proceed', views.proceed, name='proceed'), #path for the form button submit
]
```

Views.py

```
from django.shortcuts import render
from django.http import HttpResponse
from .models import customerReview
# Create your views here.
def generate(request):
    return render(request,'template1.html')
```

```
def proceed(request):
    """This function is for taking input from the form fields and to create a new user revie
w in thr row of table database"""
    #cusre=customerReview()
    #cusre.bran_name="nokia"
    use_name=str(request.GET["user_name"])
    br_name=str(request.GET["brand_name"])
    pro_name=str(request.GET["product_name"])
    revi_text=str(request.GET["review_text"])
    customerReview.objects.create(user_name=use_name,bran_name=br_name,produ_name=pro_name,r
eview_data=revi_text) #command that creates the new user row for product review in the datab
ase
    cusres=customerReview.objects.all() #the operation thta calls the all rows in table
    return render(request,'products.html',{'dests':cusres})
```

TEMPLATES

Products.html

```
<body bgcolor="yellow">
{% block content %}


<h1> List of All Product names and reviews given in form </h1>

<br>
{% for dest in dests %}
<h2>{{dest.produ_name}}</h2>  <p> </p>> <p style="color:rgb(112, 9, 5)">{{dest.review_d
ata}}</p><br> <br>
<br>
<hr>
{% endfor %}
{% endblock %}
</body>
```

Template1.html

```
<form action="proceed">
    Enter your name:<input type="text," name="user_name" placeholder="customer name"><br>
    <br>
    Enter the name of your brand:<input type="text" name="brand_name" placeholder="type he
re"><br>
    <br>
    Enter the name of your product:<input type="text" name="product_name" placeholder="typ
e here"><br>
    <br>
    Enter your review:<input type="text" name="review_text" placeholder="type here"><br>
    <br>

    <input type="submit",name="Add your review">
</form>
```