

1. Create project (flipkart), change directory and open project in vs code

```
\DJANGO T417 203>django-admin startproject flipkart
\nDJANGO T417 203>cd flipkart
\nDJANGO T417 203\flipkart>code .
```

2. Create application (product)

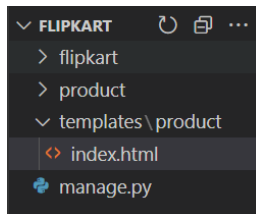
```
\DJANGO T417 203\flipkart>python manage.py startapp product
```

3. Register application in settings.py

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

4. Create templates folder at project level

- a. Create product folder in templates folder
- b. Create index.html file in product folder



5. Register templates folder in settings.py

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR/'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
```

6. Create database – flipkart from workbench

```
SQL File 6" trigger SQL File 9" SQL File 5" x
1 create database flipkart;
2
```

7. Add database details in settings.py (NAME,USER,PASSWORD,HOST,PORT)

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'flipkart',
        'USER': 'root',
        'PASSWORD': 'root',
        'HOST': 'localhost',
        'PORT': '3306'
    }
}
```

8. Runserver : if you are not getting any error then django is connected with database

```
C:\TEJAS KASARE (Very imp folder)\my notes\Batch Wise\DJANGO T417 203\flipkart>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s)
: admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
April 03, 2024 - 11:51:37
Django version 4.2.6, using settings 'flipkart.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

9. Create Product model class in models.py

```
models.py X
product > models.py > ...
1  from django.db import models
2
3  # Create your models here.
4  class Product(models.Model):
5      id = models.AutoField(primary_key=True)
6      name = models.CharField(max_length=50)
7      description = models.CharField(max_length=150)
8      price = models.IntegerField()
9
10 def __str__(self):
11     return self.name + " added to the table"
12
```

10. Register Product model class in admin.py

```
admin.py X
product > admin.py > ...
1  from django.contrib import admin
2  from product.models import Product
3
4  # Register your models here.
5  class ProductAdmin(admin.ModelAdmin):
6      list_display = ['id', 'name', 'description', 'price']
7
8  admin.site.register(Product, ProductAdmin)
9
```

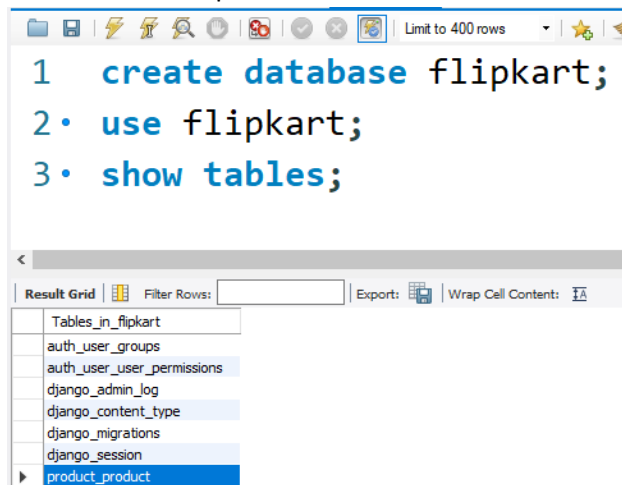
11. Covert model class into respective sql query : makemigrations

>python manage.py makemigrations

12. To execute query created by makemigrations : migrate

>python manage.py migrate

13. Check tables in flipkart database



The screenshot shows a database management interface. At the top, there are icons for file operations and a toolbar. Below the toolbar, three SQL commands are listed:

```
1 create database flipkart;
2 use flipkart;
3 show tables;
```

Below the commands, there is a section labeled "Result Grid" with a search bar and an "Export" button. A list of tables is displayed, including:

- Tables_in_flipkart
- auth_user_groups
- auth_user_user_permissions
- django_admin_log
- django_content_type
- django_migrations
- django_session
- product_product

14. Create superuser

```
C:\TEJAS KASARE (Very imp folder)\my notes\Batch Wise\DJANGO T417 203\flipkart>python manage.py createsuperuser
Username (leave blank to use 'admin'): tejas
Email address: tejas@gmail.com
Password:
Password (again):
Superuser created successfully.

C:\TEJAS KASARE (Very imp folder)\my notes\Batch Wise\DJANGO T417 203\flipkart>
```

15. Runserver and Use above credentials to login admin panel <http://127.0.0.1:8000/admin>

16. Add some products

Add product

Name:

Description:

Price:

✓ The product "iphone 15 added to the table" was added successfully.

Select product to change ADD PRODUCT +

Action: 0 of 1 selected

<input type="checkbox"/>	ID	NAME	DESCRIPTION	PRICE
<input type="checkbox"/>	1	iphone 15	good phone	1500

1 product

17. Create view – show_products() in product > views.py

- Write code to fetch all products from Product model
- Send fetched products to index.html

```
views.py X
product > views.py > ...
1 from django.shortcuts import render
2 from product.models import Product
3 # Create your views here.
4 def show_product(request):
5     data = {}
6     all_products = Product.objects.all()
7     data['products'] = all_products
8     data['name'] = "Raj"
9     return render(request, 'product/index.html', context=data)
10
```

```
index.html X
templates > product > index.html
6 <body>
7 <h1>Welcome {{name}},</h1>
8 <table border="1" cellpadding=10 cellspacing=10 align='center'>
9     <tr>
10         <th>ID</th>
11         <th>NAME</th>
12         <th>DESCRIPTION</th>
13         <th>PRICE</th>
14     </tr>
15     {% for product in products %}
16     <tr>
17         <td>{{product.id}}</td>
18         <td>{{product.name}}</td>
19         <td>{{product.description}}</td>
20         <td>{{product.price}}</td>
21     </tr>
22     {% endfor %}
23 </table>
24 </body>
```

18. Create url at application level and add it to the project level (urls.py)

- a. Create product_urls.py in product
- b. Code
- c. Register product_urls.py in urls.py (project level)

```
product_urls.py x
product > product_urls.py > ...
1 from django.urls import path
2 from product import views
3 urlpatterns = [
4     path('get/', views.show_product),
5 ]

urls.py x
flipkart > urls.py > ...
17 from django.contrib import admin
18 from django.urls import path, include
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('product/', include('product.product_urls')),
23 ]
```

19. Runserver and get Output

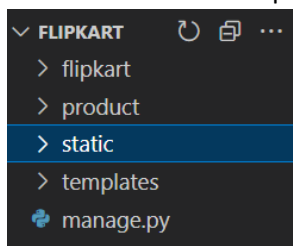


Welcome Raj,

ID	NAME	DESCRIPTION	PRICE
1	iphone 15	good phone	1500
2	LG smart TV	good tv	2900

20. Working with static files (CSS and Images)

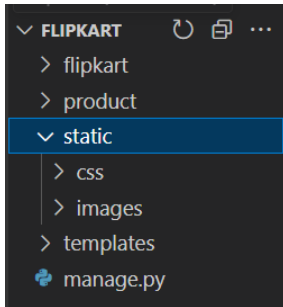
- a. Create static folder at project level (like we created templates)



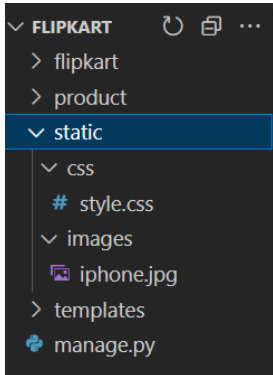
- b. Register static folder in settings.py

```
settings.py x
flipkart > settings.py > ...
122
123 STATIC_URL = 'static/'
124 STATICFILES_DIRS = [BASE_DIR/'static']
125
```

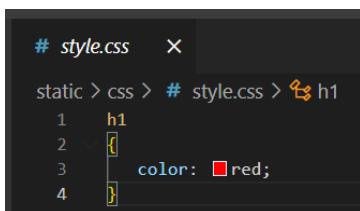
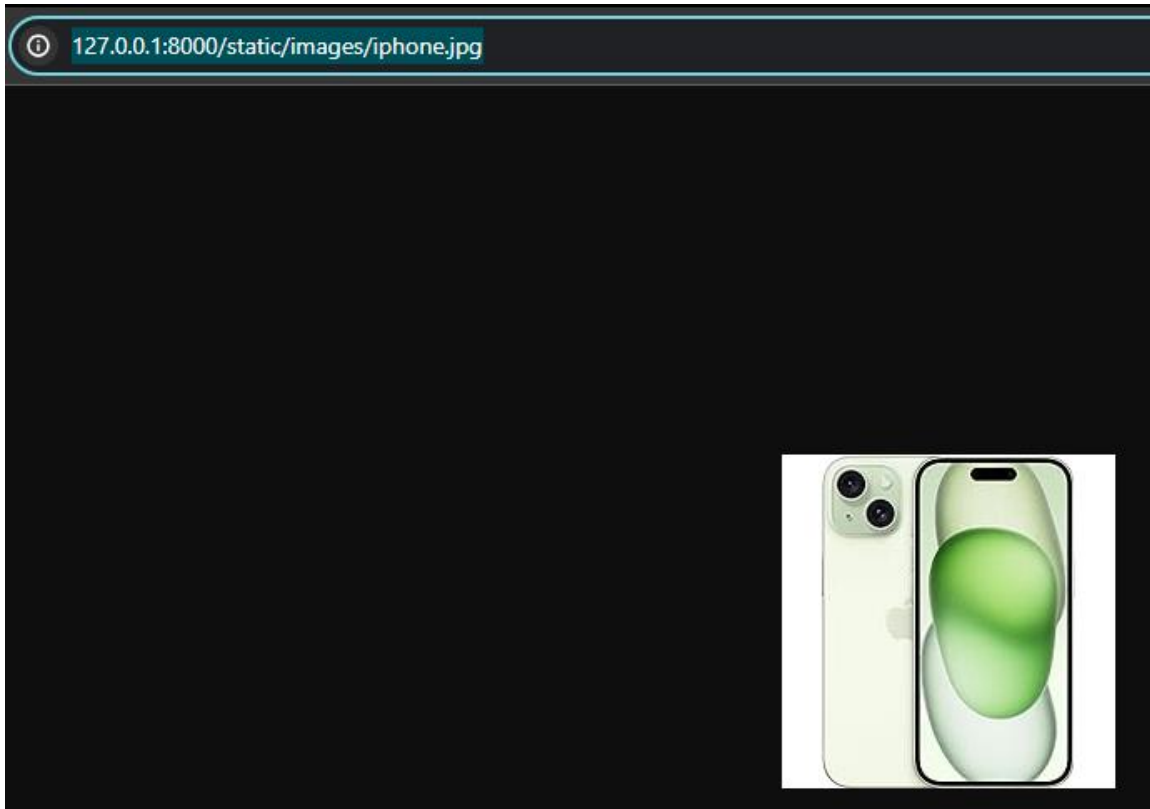
- c. Create css and images folder in static folder



- d. create style.css in above css folder and add one image in images folder



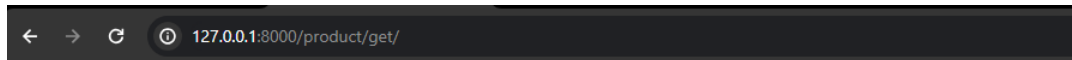
- e. you can get above image through url on the browser as –



- f. configure style.css in display image in index.html

```
index.html x
templates > product > index.html > ...
1  DOCTYPE html>
2  {% load static %}
3  <html lang="en">
4  <head>
5      <title>Document</title>
6      <link rel="stylesheet" href="{% static 'css/style.css' %}">
7  </head>
8  <body>
9      <h1>Welcome {{name}},</h1>
10     
11     <table border="1" cellpadding=10 cellspacing=10 align='center'>
```

21. output



Welcome Raj,



ID	NAME	DESCRIPTION	PRICE
1	iphone 15	good phone	1500
2	LG smart TV	good tv	2900