



MYSORE UNIVERSITY SCHOOL OF ENGINEERING

Manasagangotri campus, Mysuru-
570006 (Approved by AICTE, New
Delhi)



UNIVERSITY OF MYSORE

Full Stack Development(21CD71) Assessment Report On:
“E-Commerce Product Management System ”

Under the guidance :
Mr. Karthik M N
Assistant Professor,
Department of Computer Science &
Design,
MUSE.

Submitted by:
Samrudh ML
Reg No : 21SECD37

Introduction:

This E-Commerce Product Management System includes:

- Product Model: Contains fields like name, description, price, stock, and category.
- Category Model: Establishes a One-to-Many relationship with the Product model.
- Django Admin Integration: Enables product and category management through the admin panel.
- Product Listing with ListView: Displays a dynamic list of products on the website.
- Database Migrations: Introduces a discount field in the Product model after initial development.

Project overview:

ecommerce/

```
| — manage.py
| — db.sqlite3
| — ecommerce/
|   | — __init__.py
|   | — settings.py
|   | — urls.py
|   | — asgi.py
|   | — wsgi.py
| — products/
|   | — __init__.py
|   | — admin.py
|   | — apps.py
|   | — models.py
|   | — tests.py
|   | — views.py
|   | — urls.py
|   | — templates/
|   |   | — products/
```

```
| | | |— category_product_list.html  
|— fsdlab/ (Virtual Environment)
```

Detailed steps Implementation:

Step 1: Install Django and Create a Virtual Environment

Create a virtual environment

```
>>python -m venv fsdlab
```

Activate the virtual environment

```
>>fsdlab\Scripts\activate
```

Install Django

```
>>pip install Django
```

Step 2: Create a Django Project

Run the following command to create a Django project:

```
>>django-admin startproject ecommerce
```

```
>>cd ecommerce
```

Step 3: Create a Django App

```
>>python manage.py startapp products
```

Step 4: Configure settings.py

Open ecommerce/settings.py and add 'products' to *INSTALLED_APPS*

Step 5: Create the Student Model:

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

```
class Category(models.Model):
```

```
    name = models.CharField(max_length=100)
```

```
    description = models.TextField(blank=True)
```

```

    def __str__(self):
        return self.name

class Product(models.Model):
    name = models.CharField(max_length=100)
    description = models.TextField()
    price = models.DecimalField(max_digits=10, decimal_places=2)
    stock = models.PositiveIntegerField()
    category = models.ForeignKey(Category, related_name='products',
on_delete=models.CASCADE)
    discount = models.DecimalField(max_digits=5, decimal_places=2, default=0.00)

    def __str__(self):
        return self.name

```

Run migrations to apply the model:

```
>>python manage.py makemigrations
```

```
>>python manage.py migrate
```

Step 6: Register the Model in Django Admin:

In `products/admin.py`:

```

from django.contrib import admin
from .models import Product, Category
@admin.register(Category)
class CategoryAdmin(admin.ModelAdmin):
    list_display = ('name',)
    search_fields = ('name', 'description')
@admin.register(Product)
class ProductAdmin(admin.ModelAdmin):
    list_display = ('name', 'price', 'stock', 'category')
    list_filter = ('category',)
    search_fields = ('name', 'description')

```

Step 7: Create Views for ecommerce:

In **products/views.py**

```
from django.views.generic import ListView
from .models import Category
class CategoryProductListView(ListView):
    model = Category
    template_name = 'products/category_product_list.html' # Specify your template name
    context_object_name = 'categories'
```

Step 8: Configure URLs:

Create **products/urls.py**

```
from django.urls import path
from .views import CategoryProductListView
urlpatterns = [
    path("", CategoryProductListView.as_view(), name='category-product-list'),
]
```

Link the **products** app to the project's main **urls.py** in **ecommerce/urls.py**

```
from django.contrib import admin
from django.urls import path, include
from django.shortcuts import redirect

def redirect_to_products(request):
    return redirect('/products/')
urlpatterns = [
    path('admin/', admin.site.urls),
    path('products/', include('products.urls')),
    path("", redirect_to_products), # Redirect homepage to products
]
```

Step 9: Create HTML Templates:

Inside **products/templates/products/** create files naming:
category_product_list.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Products List</title>

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"

</head>

<body>

    <div class="container">

        <h1 class="my-4 text-center">Products List</h1>

        {% for category in categories %}

            <h2 class="my-4">{{ category.name }}</h2>

            <div class="row">

                {% for product in category.products.all %}

                    <div class="col-md-4">

                        <div class="card product-card">

                            <div class="card-body">

                                <h5 class="card-title">{{ product.name }}</h5>

                                <p class="card-text">{{ product.description|truncatewords:20 }}</p>

                                <p class="card-text"><strong>Price:</strong> ${{ product.price }}</p>

                                <p class="card-text"><strong>Stock:</strong> {{ product.stock }}</p>

                                <p><strong>Discount:</strong> {{ product.discount }}%</p>

                            </div>

                        </div>

                    </div>

                {% empty %}

                </div>

            </div>

        {% empty %}

    </div>
```

```
<div class="col-12">

    <p class="text-center">No products available in this category.</p>

</div>

{% endfor %}

</div>

{% empty %}

    <p class="text-center">No categories available.</p>

{% endfor %}

</div>

</body>

</html>
```

Step 10: Create a Superuser for Admin Panel:

```
>>python manage.py createsuperuser
```

Step 11: Run the Django Development Server

```
>>python manage.py runserver
```

Visit **127.0.0.1:8000/admin** and add products

Visit **127.0.0.1:8000/products** to see the final output

Conclusion

You have successfully created a **Django ecommerce product management system** with:

Product Model: A structured model with essential fields like name, description, price, stock, and category for efficient product management.

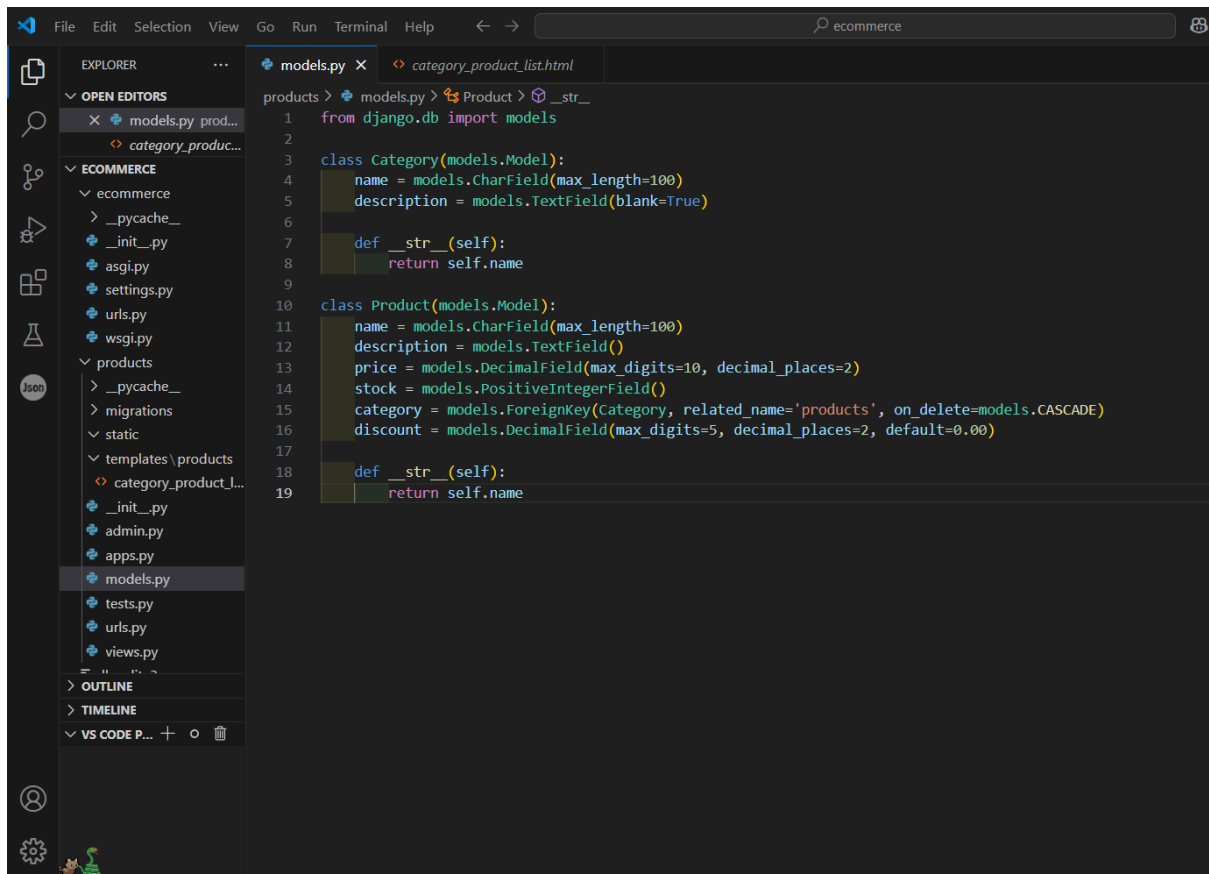
Category Model: Establishes a One-to-Many relationship with the Product model, ensuring organized product classification.

Django Admin Integration: Provides an intuitive admin panel for seamless product and category management.

Product Listing with ListView: Implements Django's ListView to dynamically display products on the website.

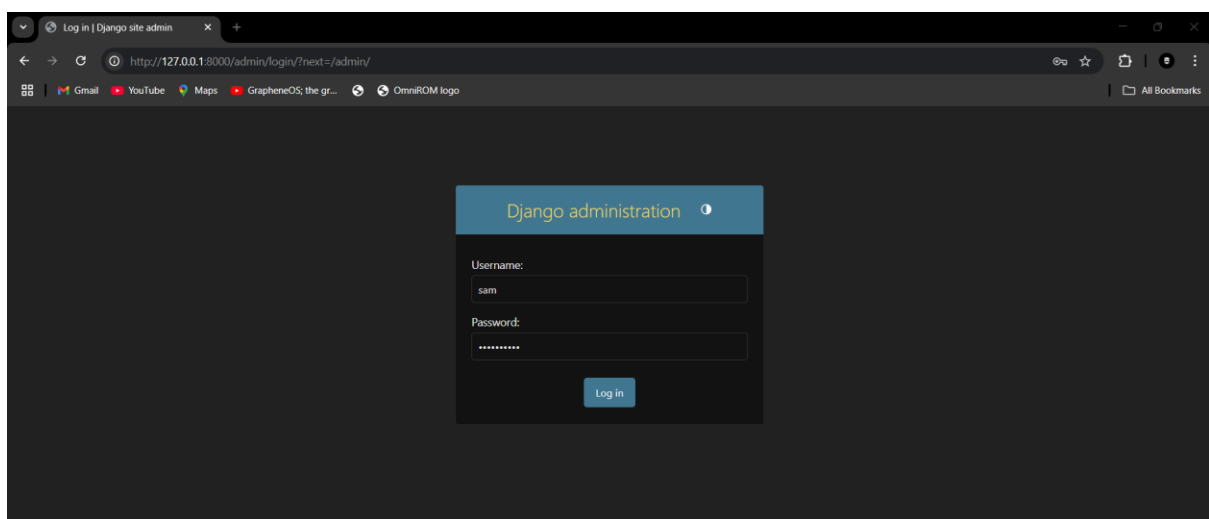
Database Migrations: Enables database schema evolution by adding a discount field to the Product model after initial development.

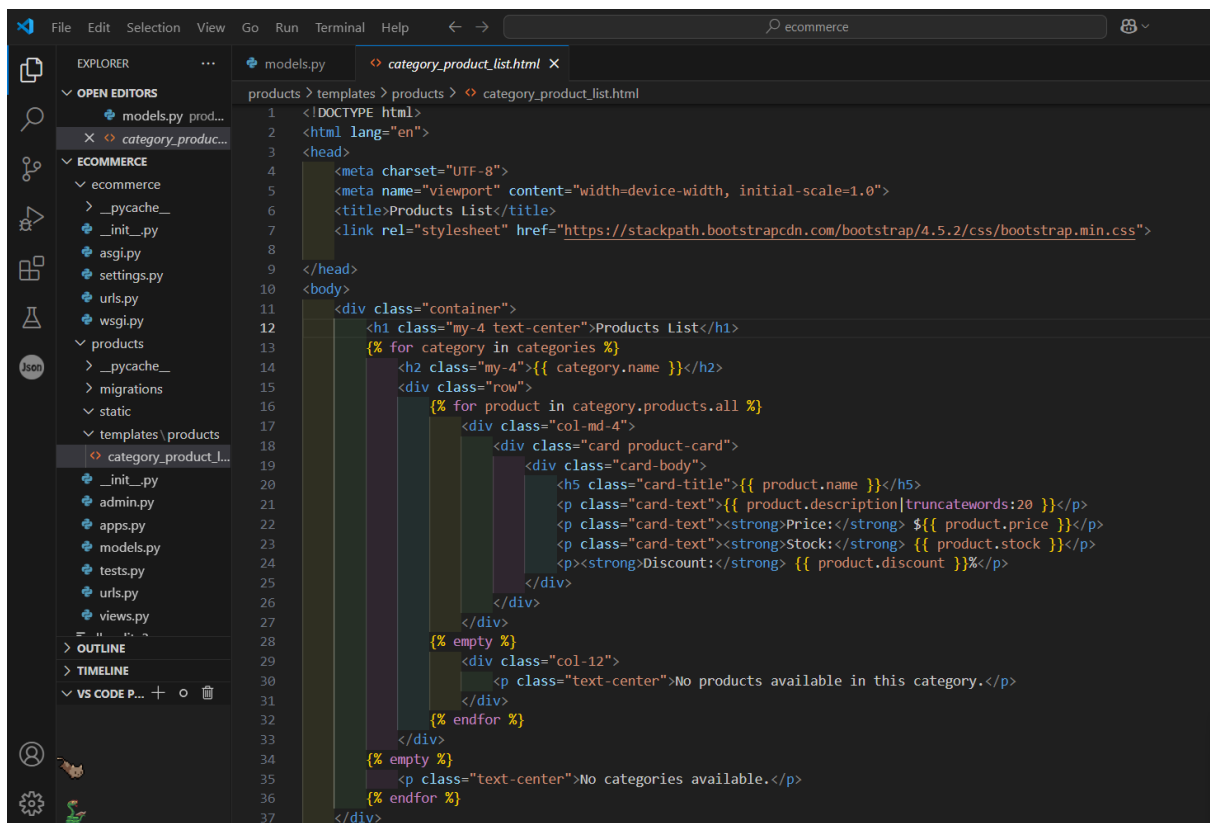
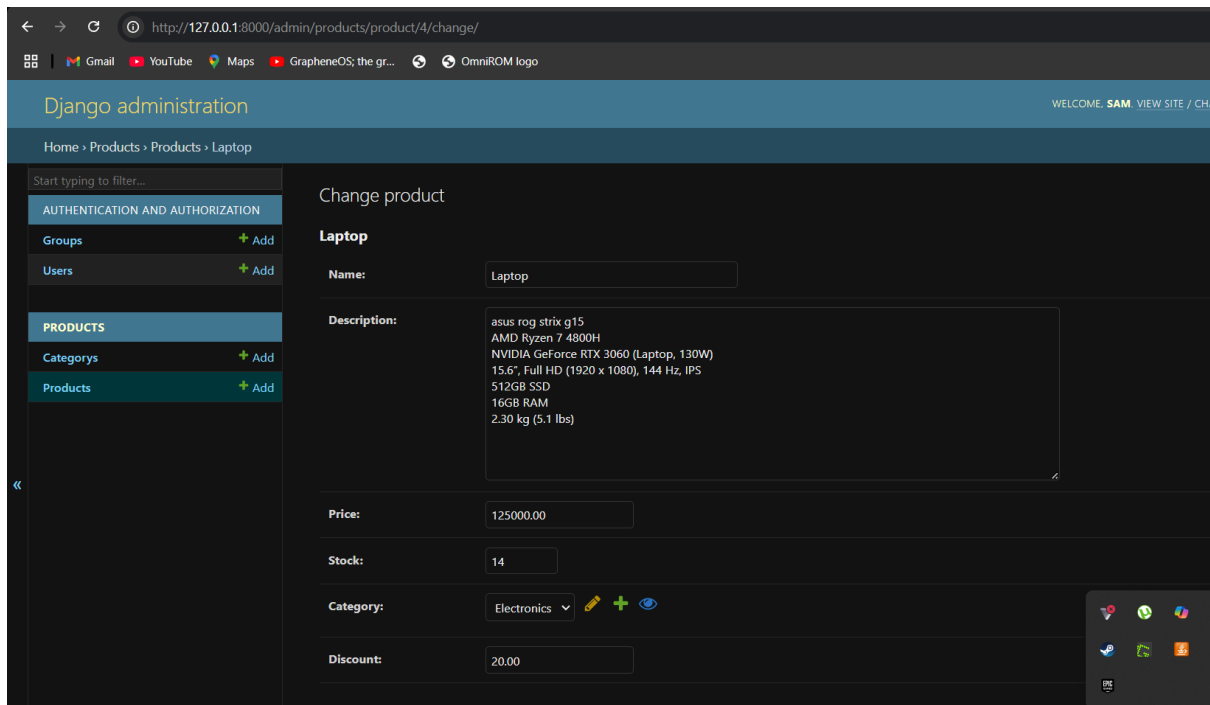
Screenshots:



A screenshot of the Visual Studio Code editor interface. The Explorer sidebar on the left shows a project structure with folders 'ecommerce' and 'products'. The 'products' folder is expanded, showing files like 'category_product_list.html', 'models.py', 'tests.py', 'urls.py', and 'views.py'. The 'models.py' file is open in the editor, showing the following Python code:

```
1 from django.db import models
2
3 class Category(models.Model):
4     name = models.CharField(max_length=100)
5     description = models.TextField(blank=True)
6
7     def __str__(self):
8         return self.name
9
10 class Product(models.Model):
11     name = models.CharField(max_length=100)
12     description = models.TextField()
13     price = models.DecimalField(max_digits=10, decimal_places=2)
14     stock = models.PositiveIntegerField()
15     category = models.ForeignKey(Category, related_name='products', on_delete=models.CASCADE)
16     discount = models.DecimalField(max_digits=5, decimal_places=2, default=0.00)
17
18     def __str__(self):
19         return self.name
```





Products List

Electronics

Smartphone samsung galaxy s69 1gb ram 8gb storage no back or front camera battery lasts for 10 -15 minutes Price: \$10000.00 Stock: 10 Discount: 20.00%	Laptop asus rog strix g15 AMD Ryzen 7 4800H NVIDIA GeForce RTX 3060 (Laptop, 130W) 15.6", Full HD (1920 x 1080), ... Price: \$125000.00 Stock: 14 Discount: 20.00%
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

clothes

shirt plain shirt allen solly blue Price: \$1299.00 Stock: 78 Discount: 15.00%	Pants Levi's denim jeans all sizes available Price: \$500.00 Stock: 34 Discount: 23.00%
------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------