

E-Commerce Product Management System Implementation Report

1. Introduction

This report details the step-by-step implementation of an E-Commerce Product Management System using Django.

The system allows managing product listings and categories via the Django Admin interface and displays products using Django's ListView.

Additionally, a database migration was performed to introduce a discount field after initial development.

2. Step-by-Step Implementation

2.1 Setting Up Django Project and App

1. Install Django:

```
pip install django
```

2. Create a new Django project and app:

```
django-admin startproject ecommerce
```

```
cd ecommerce
```

```
django-admin startapp products
```

2.2 Define Models

```
#### models.py
```

```
```python
```

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

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

```
 name = models.CharField(max_length=255, unique=True)
```

```
 def __str__(self):
```

```
 return self.name
```

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

```
 name = models.CharField(max_length=255)
```

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

```
 price = models.DecimalField(max_digits=10, decimal_places=2)
```

```
stock = models.PositiveIntegerField()

category = models.ForeignKey(Category, on_delete=models.CASCADE, related_name="products")
```

```
def __str__(self):
 return self.name
```

```
...
```

### ### 2.3 Register Models in Django Admin

#### admin.py

```
```python
```

```
from django.contrib import admin
from .models import Product, Category
```

```
@admin.register(Product)
class ProductAdmin(admin.ModelAdmin):
    list_display = ('name', 'price', 'stock', 'category')
    search_fields = ('name', 'category__name')
    list_filter = ('category',)
```

```
@admin.register(Category)
class CategoryAdmin(admin.ModelAdmin):
    list_display = ('name',)
...

```

2.4 Apply Migrations

```
```sh
python manage.py makemigrations
python manage.py migrate
...

```

### ### 2.5 Implement Product List View

#### views.py

```
```python
```

```
from django.views.generic import ListView
```

```
from .models import Product
```

```
class ProductListView(ListView):
```

```
    model = Product
```

```
    template_name = "products/product_list.html"
```

```
    context_object_name = "products"
```

```
...
```

```
#### templates/products/product_list.html
```

```
```html
```

```
{% extends "base.html" %}
```

```
{% block content %}
```

```
<h2>Product List</h2>
```

```

```

```
 {% for product in products %}
```

```

```

```
 {{ product.name }} - {{ product.price }} USD

```

```
 Category: {{ product.category.name }}

```

```
 Stock: {{ product.stock }}
```

```

```

```
 {% endfor %}
```

```

```

```
{% endblock %}
```

```
...
```

```
2.6 Configure URL Patterns
```

```
urls.py
```

```
```python
```

```
from django.urls import path
```

```
from .views import ProductListView
```

```
urlpatterns = [
```

```
    path('products/', ProductListView.as_view(), name='product-list'),
```

```
]
...
```

2.7 Add Discount Field via Migrations

Modify `models.py`

```
```python
class Product(models.Model):
 name = models.CharField(max_length=255)
 description = models.TextField()
 price = models.DecimalField(max_digits=10, decimal_places=2)
 stock = models.PositiveIntegerField()
 category = models.ForeignKey(Category, on_delete=models.CASCADE, related_name="products")
 discount = models.DecimalField(max_digits=5, decimal_places=2, default=0.00) # New Field
```
```

Run Migration

```
```sh
python manage.py makemigrations
python manage.py migrate
```
```

2.8 Run the Server and View Products

```
```sh
python manage.py runserver
```
```

Visit <http://127.0.0.1:8000/products/> to see the product list.

3. Screenshots

(Screenshots of the implemented code, web pages, and output results are attached below.)

Screenshots

127.0.0.1:8000/admin/login/?next=/admin/

Django administration

Username:

Password:

Log in

127.0.0.1:8000/admin/

Django administration

WELCOME **SATISH**. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

Change

Users

+ Add

Change

PRODUCTS

Categorys

+ Add

Change

Products

+ Add

Change

Recent actions

My actions

Bugatti Tourbillon

Product

+ Bugatti Tourbillon

Product

+ Hyper Car

Category

+ Corvette ZR1

Product

+ Sports Car

Category

+ Dodge Challenger SRT Demon 170

Product

+ Muscle Cars

Category

Bugatti Panigale V4s

Product

+ ucati Panigale V4s

Product

+ Super Bikes

Category

127.0.0.1:8000/admin/logout/

Django administration

Home

Logged out

Thanks for spending some quality time with the web site today.

[Log in again](#)

Product List

- Ducati Panigale V4s - \$6400000.00 (Stock: 500)
- Dodge Challenger SRT Demon 170 - \$18000000.00 (Stock: 5)
- Corvette ZR1 - \$20000000.00 (Stock: 5)
- Bugatti Tourbillon - \$11000000.00 (Stock: 2)

The screenshot shows a web browser at the top with the address bar displaying '127.0.0.1:8000/products/'. Below the browser is a code editor window for a Django project named 'ecommerce'. The Explorer panel on the left shows the project structure, including 'ecommerce' (with subfolders like 'products' and 'templates'), 'migrations', and 'templates'. The main editor area shows the 'manage.py' file, which contains the following code:

```
1 #!/usr/bin/env python
2 """Django's command-line utility for administrative tasks."""
3 import os
4 import sys
5
6
7 def main():
8     """Run administrative tasks."""
9     os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'ecommerce.settings')
10    try:
11        from django.core.management import execute_from_command_line
12    except ImportError as exc:
13        raise ImportError(
14            "Couldn't import Django. Are you sure it's installed and "
15            "available on your PYTHONPATH environment variable? Did you "
16            "forget to activate a virtual environment?"
17        ) from exc
18    execute_from_command_line(sys.argv)
19
20
21 if __name__ == '__main__':
22     main()
23
```

The bottom panel of the code editor shows the 'TERMINAL' output, which includes the following text:

```
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
February 21, 2025 - 22:17:11
Django version 5.1.5, using settings 'ecommerce.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

[21/Feb/2025 22:17:14] "GET / HTTP/1.1" 302 0
```