Project Completion Report – Ecom (Django E-commerce Application)

1. **Project Setup**

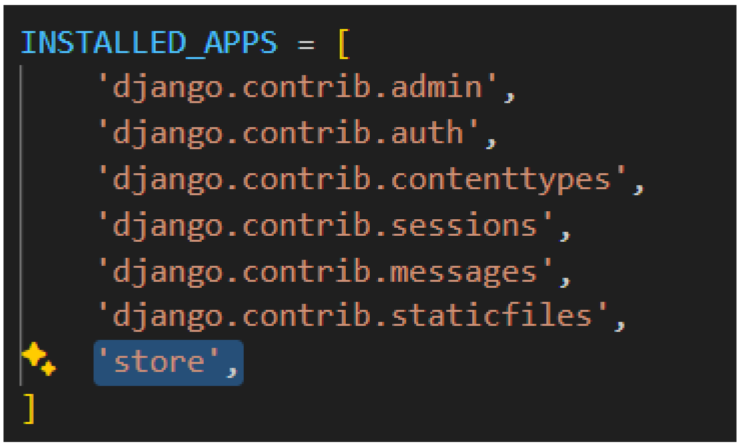
- A new Django project named Ecom was created using the command:

  django-admin startproject Ecom

- An application named store was created within the project using:

  python manage.py startapp store

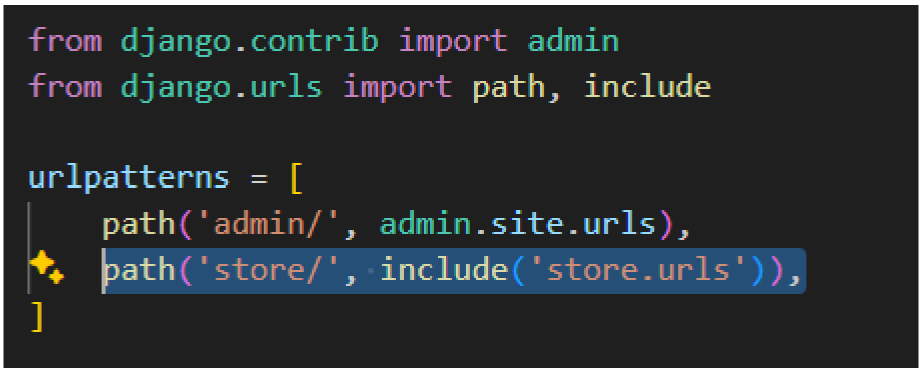
- The store app was successfully added to the INSTALLED_APPS section in settings.py.



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

2. URL Configuration

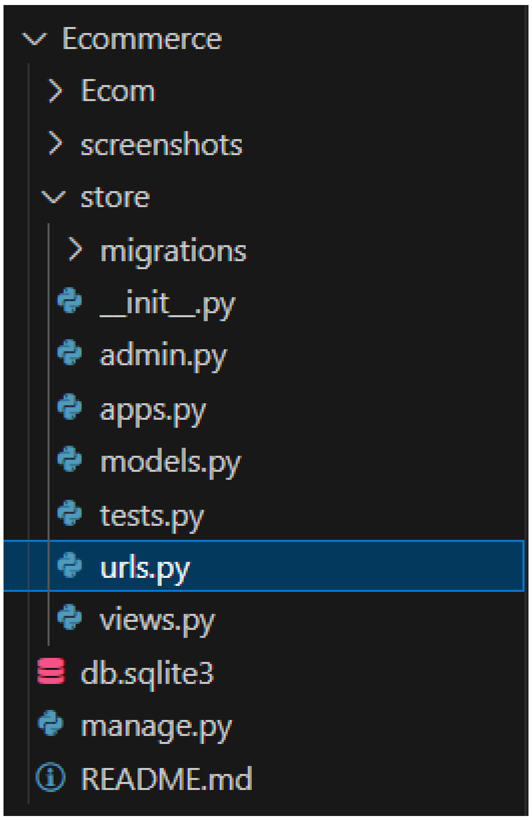
- The store app's URLs were included in the main project's URL configuration file (Ecom/urls.py).



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

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

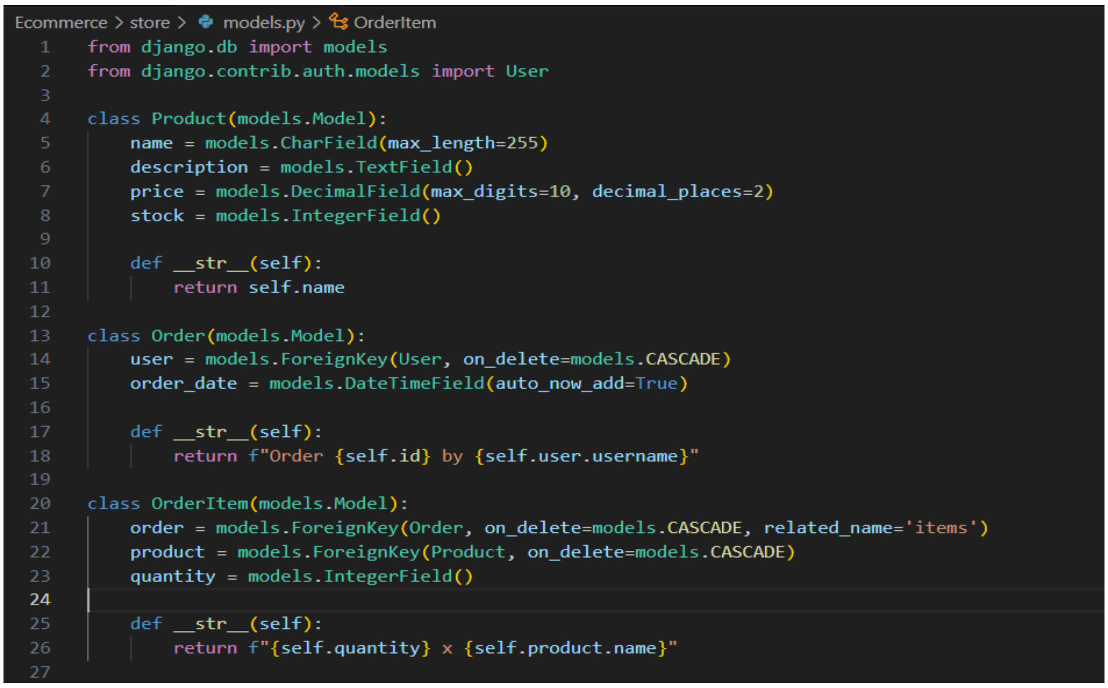
- A dedicated urls.py file was created inside the store application folder to manage its URL routing.



```
Ecommerce
  Ecom
  screenshots
  store
    migrations
    __init__.py
    admin.py
    apps.py
    models.py
    tests.py
    urls.py
    views.py
  db.sqlite3
  manage.py
  README.md
```

3. Model Creation

- Models were defined in store/models.py according to the provided schema.
- Appropriate ForeignKey relationships were established between models to ensure data integrity and relational consistency.



Ecommerce > store > models.py > OrderItem

```python
from django.db import models
from django.contrib.auth.models import User

class Product(models.Model):
    name = models.CharField(max_length=255)
    description = models.TextField()
    price = models.DecimalField(max_digits=10, decimal_places=2)
    stock = models.IntegerField()

    def __str__(self):
        return self.name

class Order(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    order_date = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return f"Order {self.id} by {self.user.username}"

class OrderItem(models.Model):
    order = models.ForeignKey(Order, on_delete=models.CASCADE, related_name='items')
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.IntegerField()

    def __str__(self):
        return f"{self.quantity} x {self.product.name}"

```

4. Database Setup

- The project was configured to use SQLite3 as the database backend.
- Database migrations were created and applied successfully to generate the necessary tables in the database.

5. Admin Interface

- All created models were registered in the Django admin site to enable easy data management.
- Admin access was verified and functional.

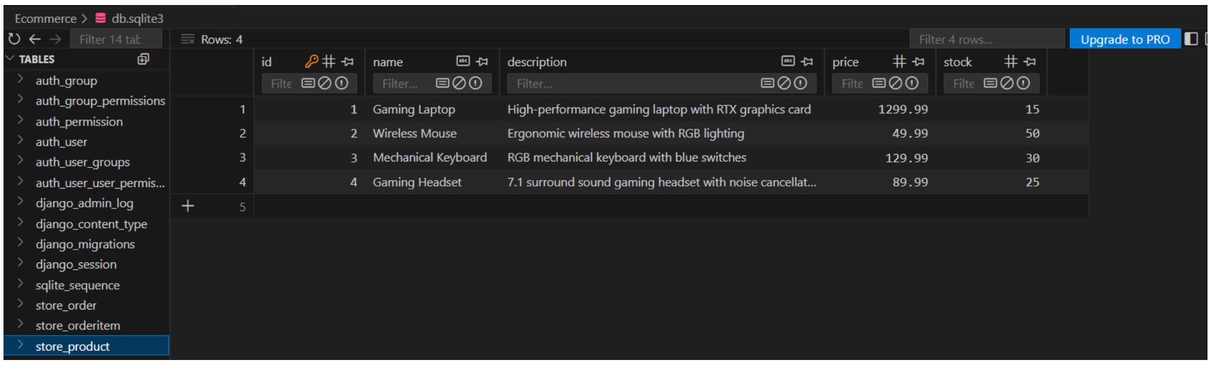
6. Testing

- Sample users, products, and orders were added through the admin interface.
- The database schema and model relationships were validated successfully based on test entries.

Database Schema

Product Table

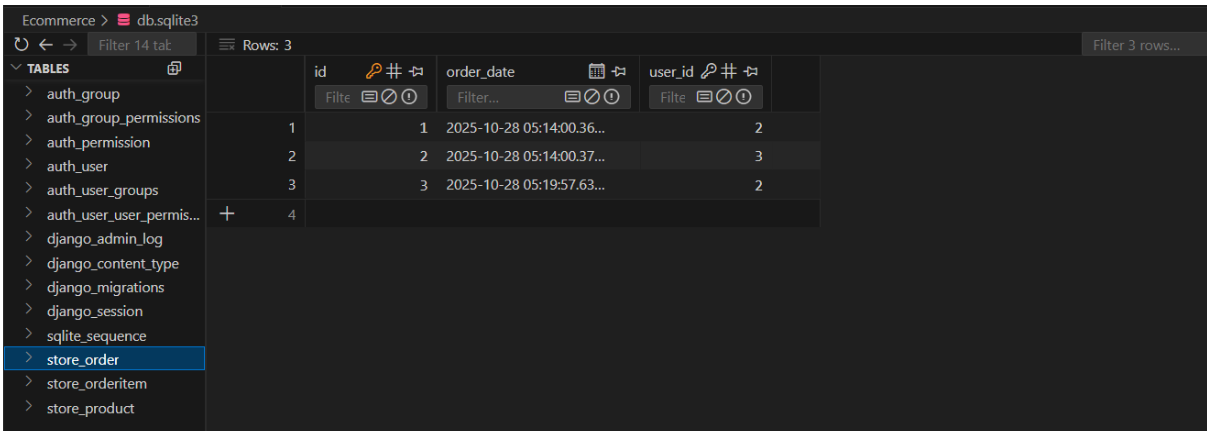
Field Name	Type	Description
id	Primary Key	Unique identifier for each product
name	String	Name of the product
description	Text	Detailed product description
price	Decimal	Price of the product
stock	Integer	Quantity available in stock



	id	name	description	price	stock
1	1	Gaming Laptop	High-performance gaming laptop with RTX graphics card	1299.99	15
2	2	Wireless Mouse	Ergonomic wireless mouse with RGB lighting	49.99	50
3	3	Mechanical Keyboard	RGB mechanical keyboard with blue switches	129.99	30
4	4	Gaming Headset	7.1 surround sound gaming headset with noise cancellat...	89.99	25
5					

Order Table

Field Name	Type	Description
id	Primary Key	Unique identifier for each order
user	Foreign Key (User)	References the user who placed the order
order_date	Timestamp	Date and time when the order was created



Ecommerce > db.sqlite3

Filter 14 tak

TABLES

- auth_group
- auth_group_permissions
- auth_permission
- auth_user
- auth_user_groups
- auth_user_user_permis...
- django_admin_log
- django_content_type
- django_migrations
- django_session
- sqlite_sequence
- store_order
- store_orderitem
- store_product

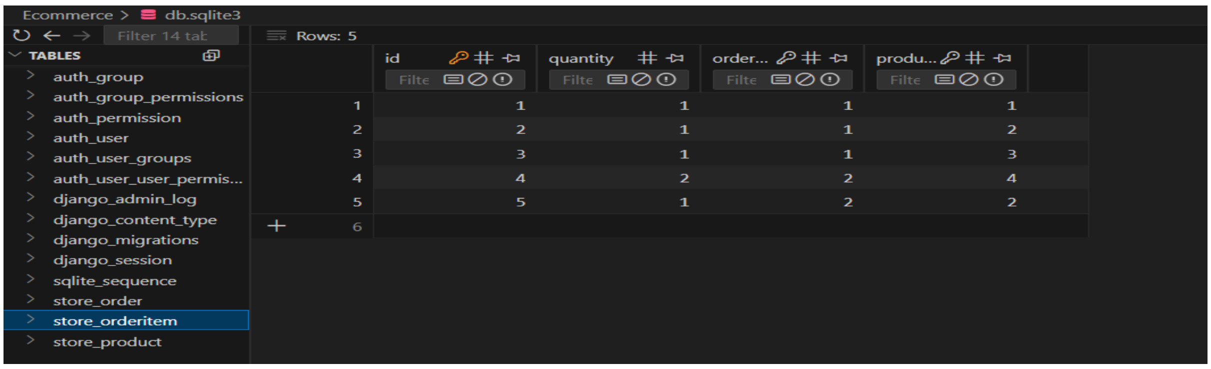
Rows: 3

Filter 3 rows...

	id	order_date	user_id
1	1	2025-10-28 05:14:00.36...	2
2	2	2025-10-28 05:14:00.37...	3
3	3	2025-10-28 05:19:57.63...	2
4			

OrderItem Table

Field Name	Type	Description
id	Primary Key	Unique identifier for each order item
order	Foreign Key (Order)	References the related order
product	Foreign Key (Product)	References the purchased product
quantity	Integer	Quantity of the product ordered



Ecommerce > db.sqlite3

Filter 14 tak

TABLES

- auth_group
- auth_group_permissions
- auth_permission
- auth_user
- auth_user_groups
- auth_user_user_permis...
- django_admin_log
- django_content_type
- django_migrations
- django_session
- sqlite_sequence
- store_order
- store_orderitem
- store_product

Rows: 5

	id	quantity	order...	produ...
1	1	1	1	1
2	2	1	1	2
3	3	1	1	3
4	4	2	2	4
5	5	1	2	2
6				