Jewelry Shop E-Commerce Website

This is a simple e-commerce website built with Django (Python) for a jewelry shop, but you can customize it for any type of e-commerce. It integrates PayPal as a payment processor. Users can browse products, add them to their cart, and make purchases using PayPal or credit/debit cards.

# Features of the Project

A. Admin Users Can:

- Manage Categories (Add, Update, Filter, and Delete)  
- Manage Products (Add, Update, Filter, and Delete)  
- Manage Users (Update, Filter, and Delete)  
- Manage Orders (View and Process)

B. Non-Registered Users Can:

- View Products (Filter based on categories)  
- Explore Product Details and Related Products

C. Registered Users Can:

- All features of non-registered users  
- Add products to the Cart  
- Pay with PayPal or Debit/Credit Card and Place an Order  
- See Order Status  
- View Order History  
- Update Profile  
- Change Password  
- Reset Password

# Prerequisites

Before you begin, ensure you have the following installed:  
- Python 3.x  
- pip (Python package installer)  
- Django  
- PayPal Developer Account (for payment processing)

# Installation (Local Setup)

1. Clone the Repository

Clone the repository to your local machine:  
```bash  
git clone https://github.com/yourusername/django-jewelry-shop.git  
cd django-jewelry-shop  
```

2. Create and Activate a Virtual Environment

Install `virtualenv` (if not installed):  
```bash  
pip install virtualenv  
```  
Create a virtual environment:  
For \*\*Windows\*\*:  
```bash  
python -m venv venv  
```  
For \*\*Mac/Linux\*\*:  
```bash  
python3 -m venv venv  
```  
Activate the virtual environment:  
For \*\*Windows\*\*:  
```bash  
source venv/scripts/activate  
```  
For \*\*Mac/Linux\*\*:  
```bash  
source venv/bin/activate  
```

3. Install Required Dependencies

Install the required packages from the `requirements.txt`:  
```bash  
pip install -r requirements.txt  
```

4. Update `ALLOWED\_HOSTS` in `settings.py`

In the `settings.py` file, update `ALLOWED\_HOSTS`:  
```python  
ALLOWED\_HOSTS = ['\*']  
```

5. Run the Development Server

For \*\*Windows\*\*:  
```bash  
python manage.py runserver  
```  
For \*\*Mac/Linux\*\*:  
```bash  
python3 manage.py runserver  
```  
The development server will now be running at [http://localhost:8000](http://localhost:8000).

6. Create a Superuser (Admin)

To access the Django admin panel, create a superuser account:  
For \*\*Windows\*\*:  
```bash  
python manage.py createsuperuser  
```  
For \*\*Mac/Linux\*\*:  
```bash  
python3 manage.py createsuperuser  
```

# Installation (EC2 Instance Setup)

1. Launch an EC2 Instance

Choose an \*\*Ubuntu\*\* instance type (e.g., `t2.micro`).  
Configure security groups to allow inbound traffic on port \*\*8000\*\* (for Django) and \*\*22\*\* (SSH).  
Ensure \*\*Python 3\*\* and \*\*pip\*\* are installed on the instance.

2. SSH into Your EC2 Instance

Use SSH to access the EC2 instance:  
```bash  
ssh -i /path/to/your-key.pem ubuntu@your-ec2-public-ip  
```

3. Set Up the Project on EC2

Install Virtual Environment on EC2:  
```bash  
sudo apt update  
sudo apt install python3-pip python3-dev libpq-dev  
sudo pip3 install virtualenv  
```

Clone the Repository on EC2:

```bash  
git clone https://github.com/yourusername/django-jewelry-shop.git  
cd django-jewelry-shop  
```

Create and Activate Virtual Environment on EC2:

```bash  
python3 -m venv venv  
source venv/bin/activate  
```

Install Dependencies on EC2:

```bash  
pip install -r requirements.txt  
```

Update `ALLOWED\_HOSTS` in `settings.py`

In the `settings.py` file, set:  
```python  
ALLOWED\_HOSTS = ['\*']  
```

Run the Development Server on EC2

```bash  
python3 manage.py runserver 0.0.0.0:8000  
```

Your site should now be accessible via your EC2 instance's public IP, e.g., [http://your-ec2-public-ip:8000](http://your-ec2-public-ip:8000).

4. (Optional) Set Up Gunicorn and Nginx for Production

For production deployments, it's recommended to use \*\*Gunicorn\*\* and \*\*Nginx\*\*. Follow the official Django deployment guide to set up these services on your EC2 instance.

# Login Credentials

- \*\*Admin Login\*\*: Use the superuser credentials you created.  
- \*\*User Login\*\*: Register a new user from the homepage or log in using existing credentials.

# Author

This project was developed by \*\*Usman Ghani\*\*.

# License

This project is licensed under the MIT License - see the LICENSE file for details.