EpicOutlet

E-Commerce Web Application

Project Report

Technology: Python · Django · AI-Powered Shopping Bot

March 2026

1. Abstract

EpicOutlet is a full-featured e-commerce web application built using Python and the Django framework. It allows users to browse, search, and purchase products across multiple categories such as electronics, fashion, groceries, and home items. The platform includes user authentication, a shopping cart, a favourites list, and an AI-powered shopping chatbot called Shop-Bot. Products and images are managed via Cloudinary cloud storage, and the application is deployed using Gunicorn on a cloud server. EpicOutlet is designed to provide a smooth and modern online shopping experience.

2. Overview

EpicOutlet is an online shopping platform where users can discover and buy products from different categories. The project was built as a complete web application with both a customer-facing front-end and an admin panel for managing products.

The main idea is to create a simple but powerful e-commerce site where:

- Users can register, log in, and manage their accounts.
- Products are organized into categories like Mobile, Fashion, Electronic, Home, and Grocery.
- Users can add items to their cart or save them as favourites.
- An AI chatbot (Shop-Bot) helps users find products using natural language.
- Admins can manage products, categories, and orders from a dashboard.

3. Objective

Problem Statement

Traditional e-commerce sites are often complex and difficult to navigate. Users spend too much time searching for products manually. There is also no intelligent assistant to guide shoppers in real time.

Main Goals

- Build a complete online shopping platform using Django and Python.
- Make it easy for users to find products using filters, search, and sorting.
- Add an AI-powered chatbot to help users shop using natural language conversation.
- Store product images safely in the cloud using Cloudinary.
- Provide a clean and responsive user interface that works on mobile and desktop.
- Include a REST API using django-tastypie for external access to product data.

4. Scope

Where It Can Be Used

- Small to medium online retail businesses looking for a ready-to-deploy store.
- Student or portfolio projects demonstrating full-stack Django development.
- Any seller who wants to quickly set up an online product catalog with AI assistance.

Who Can Benefit

- End users / shoppers who want a quick and smart way to find and buy products.
- Business owners who need a simple admin panel to manage their inventory.
- Developers who want to learn how to integrate AI chatbots into Django projects.
- Recruiters and clients evaluating the developer's full-stack Python skills.

5. Approach

How the Project Works

EpicOutlet follows the standard Django MVT (Model-View-Template) pattern. Here is the main workflow:

- Models: The database has tables for Categories, Products, Cart, Favourites, and Users.
- Views: Python functions handle each page request — home, product details, cart, chat, etc.
- Templates: HTML files with Django template tags render the final pages shown to the user.

AI Chatbot Logic

The Shop-Bot feature connects to the Groq AI API (using the OpenAI-compatible client). When a user types a message, the AI uses a tool called search_products to look up items in the database. The search supports filtering by category, price range, vendor, and trending status. The AI then formats the results in Markdown and sends them back to the user.

Image Storage

All product and category images are stored in Cloudinary, a cloud media platform. This keeps the server lightweight and makes image delivery fast globally.

REST API

The project exposes a REST API at /api/v1/ using django-tastypie. This allows external apps to access product, category, cart, and favourite data programmatically.

6. Tools & Technologies Used

Category	Tool / Technology	Purpose
Language	Python 3.11 / 3.13	Core programming language
Web Framework	Django 5.2	Backend MVC framework
AI / LLM	Groq API (GPT-OSS 120B)	Powers the Shop-Bot chatbot
AI Client	OpenAI Python SDK	Used to connect to Groq API
REST API	django-tastypie	Exposes product data as REST API
Database	PostgreSQL (Supabase)	Stores all application data
Cloud Images	Cloudinary	Stores and serves product images
Static Files	WhiteNoise	Serves CSS/JS files in production
Web Server	Gunicorn	Runs Django in production
Deployment	Render (Cloud)	Hosts the live application
Frontend	Bootstrap 5.3	Responsive UI components
Markdown	Marked.js	Renders chatbot responses in the browser
Icons	Font Awesome	UI icons for buttons and navigation
Admin UI	Jazzmin	Enhanced Django admin dashboard
Env Vars	python-dotenv	Loads secret keys from .env file

7. Problems Solved

- Manual Product Search: Users no longer need to click through multiple pages; they can simply ask the chatbot in plain English like "Show me phones under ₹30,000".
- Complex Filtering: The search_products tool automatically detects categories and applies filters like price range, vendor, and trending status.
- Image Hosting: By using Cloudinary, the developer avoids managing image files on the server, which is a common pain point in deployment.
- Scalable API: The REST API allows third-party integrations without additional development effort.
- Mobile Responsiveness: The app uses Bootstrap so it looks great on phones, tablets, and desktops.
- Session-Based Chat History: The chatbot remembers the last 12 messages per session, so users can ask follow-up questions naturally.
- Rate Limiting Handling: The chat service gracefully handles API rate limit errors by showing user-friendly messages.

8. Outcome / Results

After running the project, the following results are achieved:

- A fully functional online store with product listing, details, cart, and favourites.
- Users can register and log in securely to manage their shopping sessions.
- The AI chatbot successfully searches and recommends products based on user messages.
- Admins can add, edit, or hide products and categories using the Jazzmin-powered admin panel.
- Products display original and discounted prices with trending badges.
- The REST API is accessible at /api/v1/ and returns product data in JSON format.
- Images load fast from Cloudinary CDN without slowing down the server.
- The application is live and deployable on Render with Gunicorn as the web server.

9. Project Summary

EpicOutlet is a well-structured, production-ready Django e-commerce application built by Hudson Mathew. It brings together core web development skills (Python, Django, PostgreSQL, Bootstrap) with modern AI integration (Groq LLM API) and cloud services (Cloudinary, Supabase, Render).

The project demonstrates a wide range of software development skills including:

- Full-stack web development with Django MVT architecture.
- Database design with relational models (Products, Categories, Cart, Favourites).
- AI/LLM integration using tool-calling with the Groq API.
- REST API development using django-tastypie.
- Cloud deployment using Render, Cloudinary, and Supabase PostgreSQL.

Overall, EpicOutlet is a strong portfolio project that showcases the ability to build, deploy, and integrate AI into a real-world web application using Python — making it a great demonstration of full-stack and AI-assisted development skills.