

Django Ecommerce Project: NexaMart

Ujwal Sahu

TY - B.Voc AIDS

Batch 2

Roll: 34

Abstract:

NexaMart is a comprehensive e-commerce platform designed to provide users with a seamless online shopping experience. The platform offers a diverse range of products, including Mobiles, Laptops, Top Wear, and Bottom Wear, catering to various customer needs. Built using Django, NexaMart integrates robust features such as user authentication, product browsing, cart management, and order placement. Customers can register, manage their profiles, and save multiple shipping addresses for convenience. The platform also includes an AI-powered chatbot, ShopBot, which assists users by answering queries about products, prices, and categories. Additionally, Nexa Mart supports secure payment options, including PayPal integration, ensuring a smooth checkout process. With a user-friendly interface and dynamic features, Nexa Mart aims to deliver a reliable and enjoyable shopping experience.

Key Features:

1. User Authentication: Secure login and registration system for customers.
2. Product Browsing: Explore a wide range of products across multiple categories.
3. Cart Management: Add, update, and remove items from the shopping cart.
4. Order Placement: Seamless order placement with real-time status tracking.
5. Payment Integration: Secure payment options, including PayPal integration.
6. AI Chatbot: ShopBot assists users with product queries and recommendations.
7. Responsive Design: Mobile-friendly interface for a smooth user experience.
8. Profile Management: Update user details and manage multiple shipping addresses.
9. Dynamic Product Display: Products displayed with real-time pricing and discounts.
10. Order History: View past orders and their statuses.

Project structure:

NexaMart/

└─ db.sqlite3

└─ manage.py

└─ requirements.txt

└─ app/

| └─ models.py

| └─ views.py

| └─ urls.py

| └─ templates/

| | └─ app/ (HTML templates for pages like home, cart, profile, etc.)

| └─ static/

| | └─ app/ (CSS, JS, and images)

| └─ migrations/ (Database migration files)

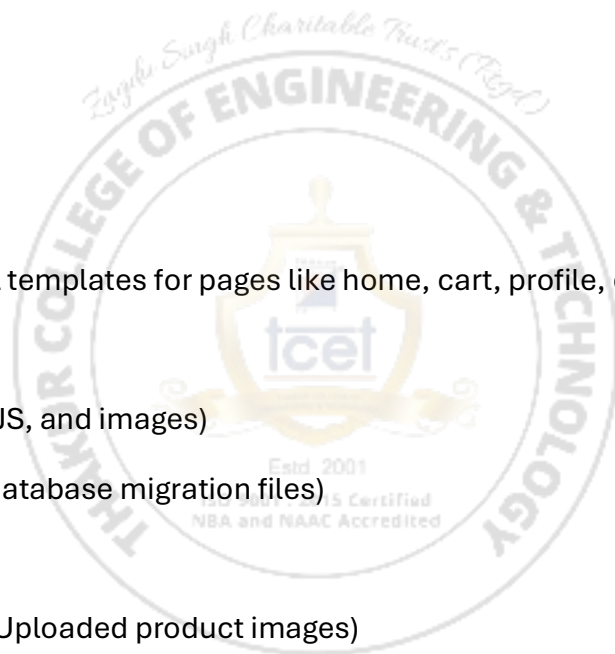
└─ media/

| └─ productimg/ (Uploaded product images)

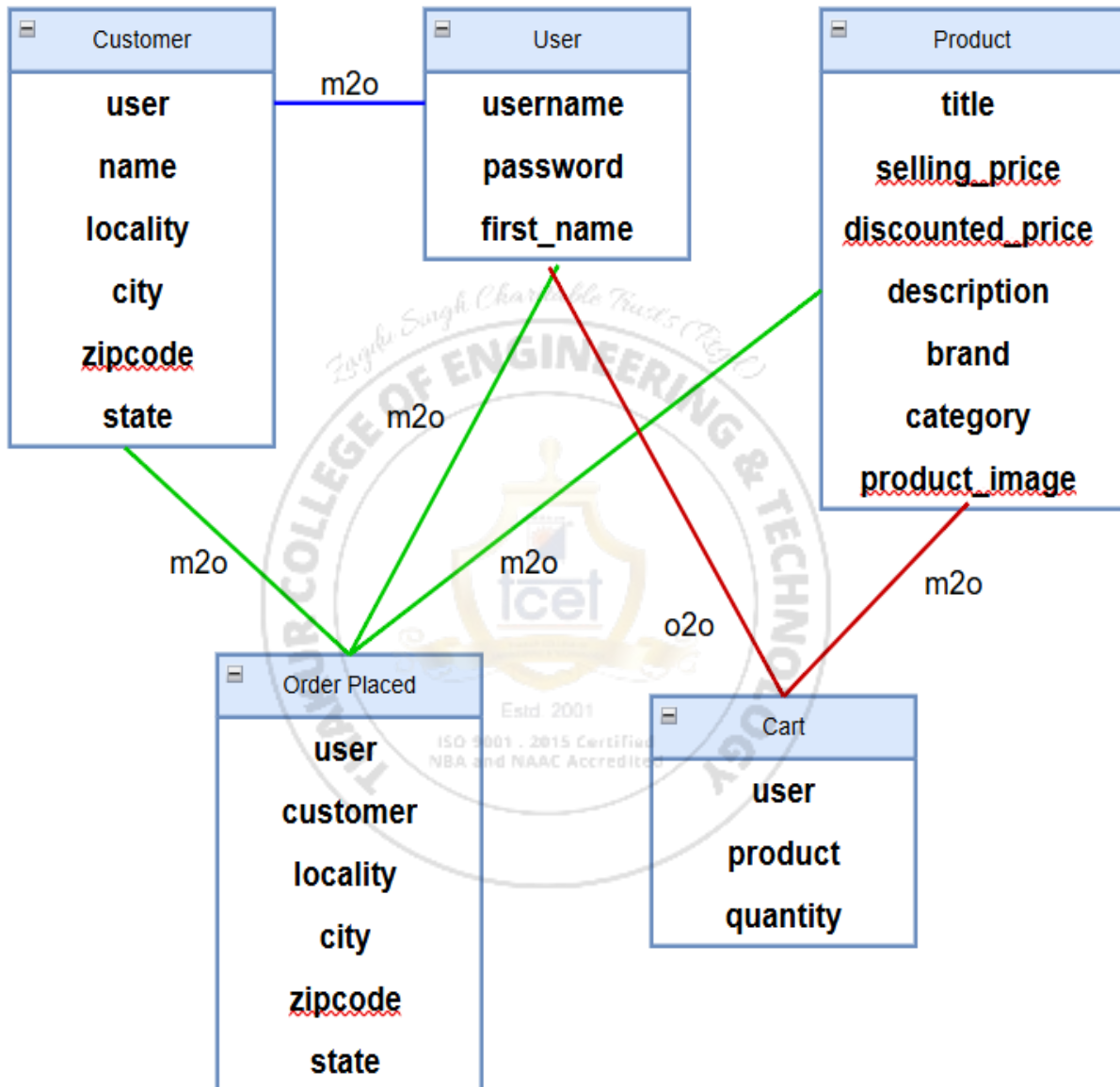
└─ shoppinglyx/

| └─ settings.py

| └─ urls.py



Tables:



1. Customer Table

Field	Type
user	ForeignKey
name	CharField
locality	CharField
city	CharField
zipcode	IntegerField
state	CharField

- Represents customer details linked to a registered user.
- **Attributes:**
 - user → FK mapping to the User who owns this customer profile.
 - name, locality, city, state, zipcode → personal + address info for deliveries.
- **Role:** Stores the delivery profile and address details of each customer.
- **Connections:**
 - Linked to **User** (m:1 → many customers can map to one user account if needed).
 - Linked to **OrderPlaced** to track which customer placed an order.

2. User Table

Field	Type
username	Primary Key
password	
first_name	

- Core authentication entity for the system.
- **Attributes:**
 - username (PK), password, first_name.
- **Role:** Holds login credentials and identity of the e-commerce platform users.
- **Connections:**
 - Owns a **Customer profile** (1:N).
 - Owns **Cart** (1:1 or 1:N depending on business rules).

- Linked to **OrderPlaced** to track which user placed orders.

3. Product Table

Field	Type
title	CharField
selling_price	FloatField
discounted_price	FloatField
description	TextField
brand	CharField
category	CharField
product_image	ImageField

- Represents the catalog of items available for purchase.
- **Attributes:**
 - title, selling_price, discounted_price, description, brand, category, product_image.
- **Role:** Provides product details for customers to browse, add to cart, and order.
- **Connections:**
 - Appears in **Cart** (N:1, multiple cart rows can point to one product).
 - Included in **OrderPlaced** (N:1, multiple orders can include the same product).

4. Cart Table

Field	Type
user	ForeignKey
product	ForeignKey
quantity	PositiveIntegerField

- Represents the **shopping cart** of each user.
- **Attributes:**
 - user (FK), product (FK), quantity.
- **Role:** Acts as a junction table between **User** and **Product**.

- **Connections:**
 - Each User can have multiple Cart rows → each row maps a product + quantity.
 - Each Product can appear in many users' carts.

5. OrderPlaced Table

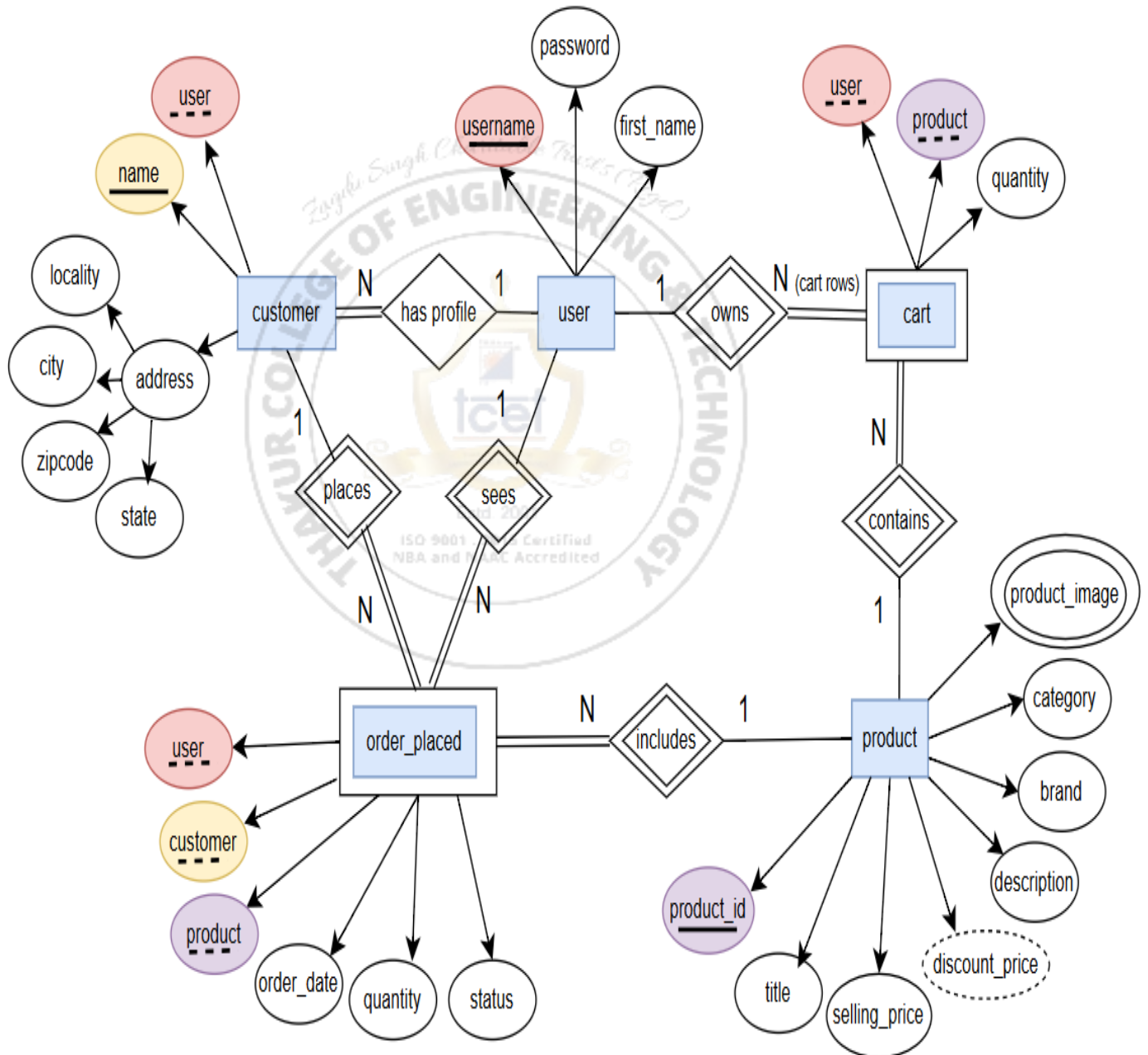
Field	Type
user	ForeignKey
customer	ForeignKey
product	ForeignKey
quantity	PositiveIntegerField
ordered_date	DateTimeField
status	CharField

- Records actual orders made by customers.
- **Attributes:**
 - user (FK), customer (FK), locality, city, state, zipcode.
- **Role:** Tracks orders with associated customer details (delivery address) and references to the buyer.
- **Connections:**
 - Linked to **User** (who placed it).
 - Linked to **Customer** (delivery details).
 - Linked to **Product** (which product(s) were ordered).

In short:

- **User** = account identity.
- **Customer** = delivery profile.
- **Product** = items to buy.
- **Cart** = temporary storage of items before purchase.
- **OrderPlaced** = finalized record of purchase + delivery details.

ER Diagram:



Entities:

1. User (Strong Entity)

- a. Attributes: username (PK), password, first_name.
- b. Strong entity (single rectangle).

2. Customer (Strong Entity)

- a. Attributes: user (FK → User), name, address (composite: locality, city, state, zipcode).
- b. Strong entity.

3. Cart (Weak Entity)

- a. Attributes: user (FK), product (FK), quantity.
- b. Weak because it depends on both User + Product (double rectangle, identifying relationship).

4. Product (Strong Entity)

- a. Attributes: product_id (PK), title, description, brand, category, product_image, selling_price, discount_price (multivalued/optional).
- b. Strong entity.

5. Order_Placed (Weak Entity)

- a. Attributes: user (FK), customer (FK), product (FK), order_date, quantity, status.
- b. Weak because it depends on User, Customer, and Product.

Relationships:

1. User — has profile — Customer

- a. Cardinality: **1 User → N Customers.**
- b. Participation: **total (double line)** on Customer side (customer cannot exist without user).
- c. Relationship: Strong.

2. User — owns — Cart

- a. Cardinality: **1 User → N Cart rows.**
- b. Participation: total on Cart side (cart cannot exist without user).
- c. Weak identifying relationship (double diamond).

3. Cart — contains — Product

- a. Cardinality: **N Cart rows → 1 Product.**
- b. Participation: total on Cart side (cart row must point to a product).
- c. Strong relationship.

4. Customer — places — Order_Placed

- a. Cardinality: **1 Customer → N Orders.**
- b. Participation: total on Order side (order cannot exist without customer).
- c. Strong relationship.

5. User — sees — Order_Placed

- a. Cardinality: **1 User → N Orders.**
- b. Participation: partial (user can exist without placing orders).
- c. Strong relationship.

6. Order_Placed — includes — Product

- a. Cardinality: **N Orders → 1 Product.**
- b. Participation: total on Order side (order must include product).
- c. Strong relationship.

User Flow for NexaMart:

Home Page (Guest User):

The user lands on the home page, where they can browse products and view categories.

Register Page:

If the user is new, they can navigate to the registration page to create an account.

Login Page:

Registered users can log in to access their personalized account.

Profile Page:

After logging in, the user can visit the profile page to manage their account.

They can add addresses (each address is treated as a customer).

Address Tab:

The user can view all their saved addresses in the address tab.

Home Page (Logged In):

Once logged in, the user can browse products and categories with personalized options.

Category Page:

Clicking on a category button shows products specific to that category.

Filter options are available to refine the product list (e.g., by brand, price range).

Product Details Page:

The user can view detailed information about a product, including images, price, and description.

Add to Cart / Buy:

The user can add products to their cart or proceed to buy directly.

Cart Page:

The user can view all items in their cart.

Options include increasing or decreasing the quantity, removing items, and viewing the total cost.

Checkout Page:

The user selects a shipping address and proceeds to checkout.

After checkout, all items in the cart are removed.

Payment (PayPal Integration):

The user completes the payment using PayPal or other integrated payment methods.

Orders Page:

After placing an order, the user can view all their orders on the orders page. Each order displays its status (e.g., Accepted, Packed, On The Way, Delivered).

Change Password Page:

The user can change their password from the profile settings.

Reset Password:

If the user forgets their password, they can reset it via email.

AI Chatbot:

The platform includes an AI-powered chatbot, ShopBot, which assists users by answering queries about products, prices, and categories. It provides a conversational interface to enhance user engagement and simplify navigation.

Admin Page:

The admin page allows administrators to manage the platform efficiently. Key functionalities include

Conclusion:

NexaMart is a robust and user-friendly e-commerce platform designed to provide a seamless shopping experience. With features like dynamic product displays, secure payment integration, and an AI-powered chatbot, it caters to modern user needs. The platform's clean UI, responsive design, and efficient backend ensure reliability and scalability. NexaMart is a comprehensive solution for online shopping, offering convenience and satisfaction to its users.

For Faculty Use

Correction Parameters	Formative Assessment [40%]	Timely completion of Practical [40%]	Attendance / Learning Attitude [20%]	
Marks Obtained				