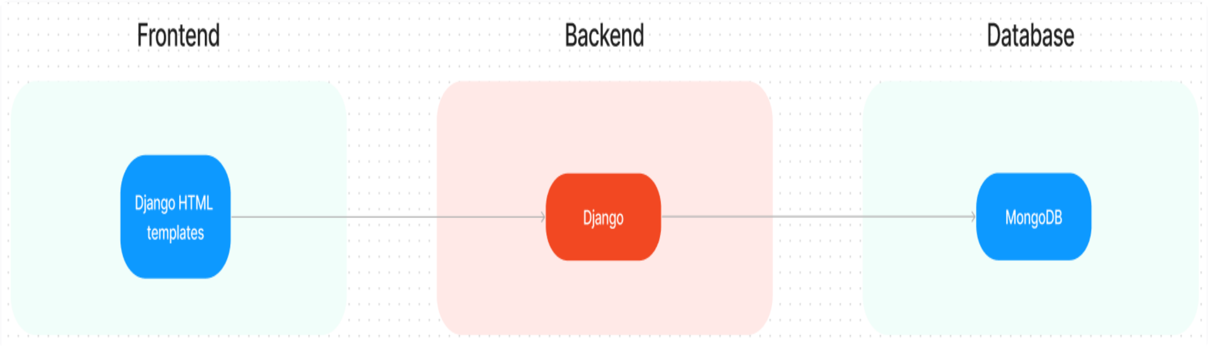
ShopEZ APP USING Django

With ShopEZ, users can effortlessly access detailed information about products, including descriptions, prices, available sizes or variations, and customer reviews. The purchasing process is quick and easy, requiring only basic customer information and payment details. Once an order is placed, users can conveniently track their purchases through the order details page. For store administrators, the platform offers an intuitive dashboard to efficiently manage product listings, monitor sales, and maintain inventory levels.

Scenario based case-study

Imagine a busy parent shopping for their family's needs. With ShopEZ, they can quickly search for products across various categories, compare options based on price and customer ratings, and complete their purchase in just a few clicks. The user can then access their order details on the go, making it easy to track shipments or modify orders as needed. Meanwhile, the store owner benefits from real-time sales data, allowing them to optimize inventory and allocate resources effectively.

TECHNICAL ARCHITECTURE

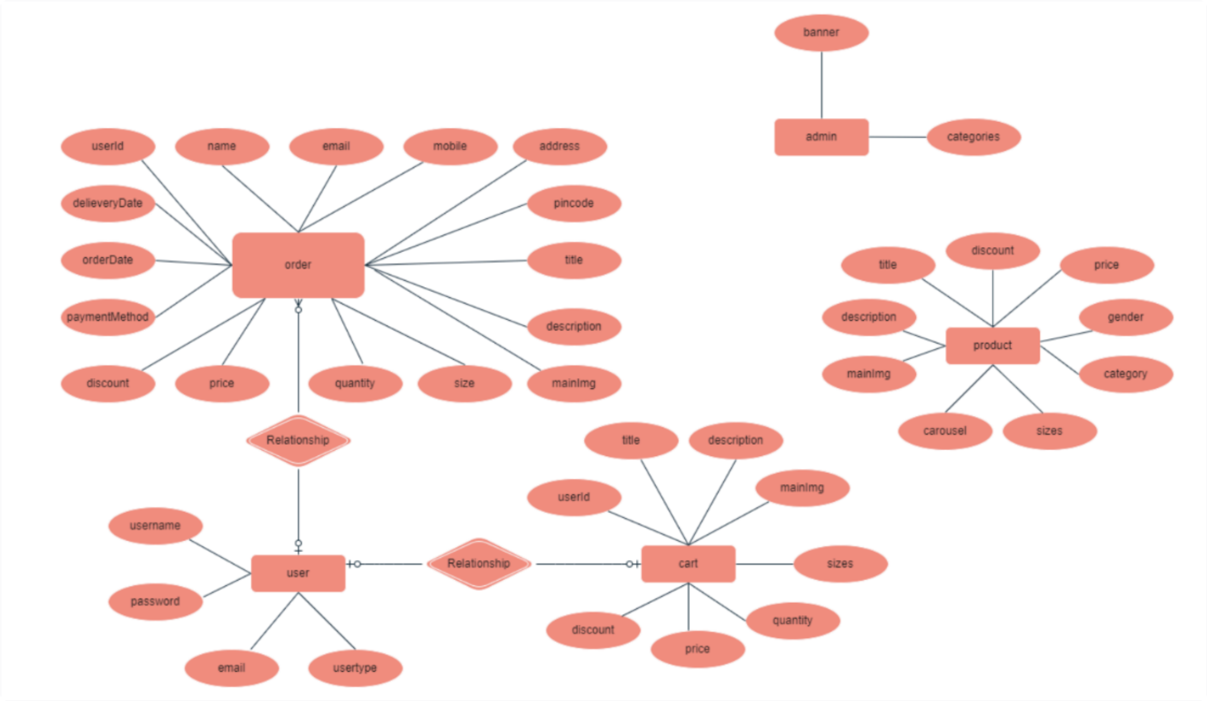


The technical architecture of our ShopEZ app follows a client-server model, where the frontend serves as the client and the backend acts as the server. The frontend utilizes the bootstrap library to establish real-time and better UI experience for any user whether it is admin, client or ordinary user working on it. On the backend side, we employ Django framework to handle the server-side logic and communication.

For data storage and retrieval, our backend relies on MongoDB. MongoDB allows for efficient and scalable storage of user data, etc. It ensures reliable and quick access to the necessary information.

Together, the frontend and backend components, along with Django and MongoDB, form a comprehensive technical architecture for our ShopEZ app. This architecture enables real-time communication, efficient data exchange, and seamless integration, ensuring a smooth and immersive booking a property and many more experience for all users.

ER DIAGRAM



The ShopEZ ER-diagram represents the entities and relationships involved in an e-commerce system. It illustrates how users, products, orders, and shopping carts are interconnected. Here is a breakdown of the entities and their relationships:

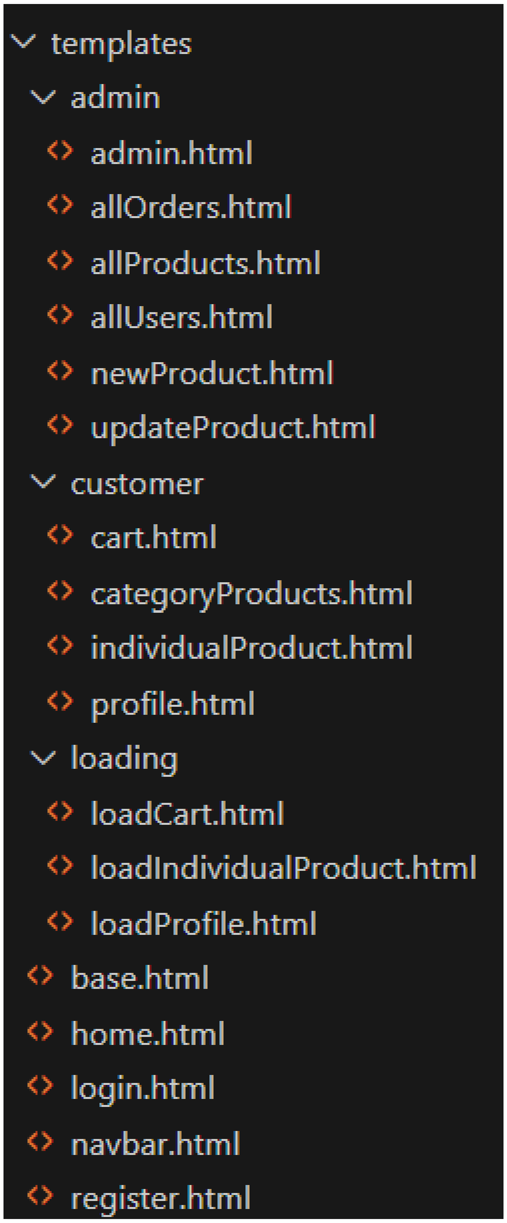
USER: Represents individuals who shop on the platform. A user can place multiple orders, add multiple products to their cart, and make multiple payments. Users can also leave reviews on products.

PRODUCT: Represents items available for purchase on the platform. Each product has details such as name, description, price, and inventory count. Products can be part of multiple orders and can be added to multiple users' carts.

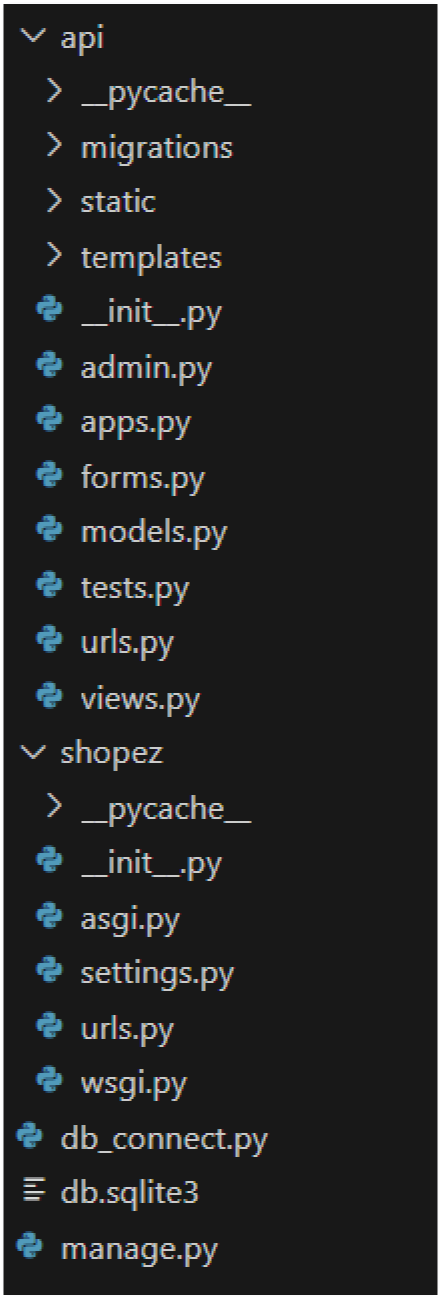
ORDERS: Represents a specific purchase made by a user. An order includes particular product details, quantities, and payment information. A user can have multiple orders, and each order can contain multiple products.

CART: Represents a user's shopping cart. It's a temporary storage of products that a user intends to purchase. A user has one cart at a time, and a cart can contain multiple products.

PROJECT STRUCTURE



```
templates
  admin
    admin.html
    allOrders.html
    allProducts.html
    allUsers.html
    newProduct.html
    updateProduct.html
  customer
    cart.html
    categoryProducts.html
    individualProduct.html
    profile.html
  loading
    loadCart.html
    loadIndividualProduct.html
    loadProfile.html
  base.html
  home.html
  login.html
  navbar.html
  register.html
```



```
api
  __pycache__
  migrations
  static
  templates
  __init__.py
  admin.py
  apps.py
  forms.py
  models.py
  tests.py
  urls.py
  views.py
shopez
  __pycache__
  __init__.py
  asgi.py
  settings.py
  urls.py
  wsgi.py
db_connect.py
db.sqlite3
manage.py
```

The first image shows the frontend content. It has a well-organized structure with dedicated folders for the API, static files (CSS and images), and templates (admin, and user). The API folder likely contains the code for the application's functionalities. The static folder stores the application's static files, such as Cascading Style Sheets (CSS) used for styling the user interface and images that may be displayed throughout the application. Finally, the templates folder contains the HTML templates that define the application's layout and user interface. These templates are used to dynamically generate the web pages that users will see.

The second image shows details about the server login. It follows a well-defined structure separating the core application logic from the overall project configuration. The application code resides in the ShopEZ directory, containing models, views, and templates specific to the ShopEZ functionality. The main project directory ShopEZ project-wide settings, URL patterns, and management tools. This separation promotes maintainability and scalability as the project grows.

Features:

Extensive Product Catalog

ShopEZ offers an extensive catalog of products, providing a wide range of categories and options for shoppers. You can easily browse through the catalog and explore different items, including detailed descriptions, high-quality images, pricing information, and customer reviews, to find the perfect products for your needs.

Add to Cart Button

Each product listing includes a convenient "Add to Cart" button. When you find an item that you'd like to purchase, simply click on the button to add it to your shopping cart. You can adjust quantities or remove items from your cart at any time before checkout.

Product Details Page

Clicking on a product will take you to its detailed page. Here, you can find comprehensive information about the item, including:

- Full product description
- Available sizes, colors, or other variations
- Price and any applicable discounts
- Customer reviews and ratings
- Related or recommended products

Secure and Efficient Checkout Process

ShopEZ ensures a secure and efficient checkout process. Your personal and payment information is handled with the utmost care, utilizing industry-standard encryption protocols. We strive to make the purchasing process as quick and hassle-free as possible, with options for guest checkout or creating an account for faster future purchases.

Order Confirmation and Tracking Page

Once you have successfully placed an order, you will receive a confirmation message. You will then be redirected to an order details page, where you can review all the relevant information about your purchase, including:

- Order number and date
- Items purchased and their quantities
- Shipping address and method
- Payment information
- Estimated delivery date

From this page, you can also track your order status as it moves through processing, shipping, and delivery stages.

User Account Dashboard:

Registered users have access to a personalized dashboard where they can:

- View order history and reorder past purchases
- Manage personal information and preferences
- Save multiple shipping addresses
- View and manage wishlist items
- Access personalized product recommendations

These features are designed to provide a seamless and enjoyable shopping experience on the ShopEZ platform, from browsing products to completing purchases and managing your account.

Pre-requisites

Here are the key prerequisites for developing a full-stack application using Django, MongoDB:

Django: Setting Up Your Project

Django is a high-level Python framework for building web applications. It streamlines the development process by providing a robust structure and handling common web development tasks.

Prerequisites:

- Python (version 3.6 or later recommended)
- pip (Python package installer)

Installation:

- Verify Python Installation: Open a terminal or command prompt and type python --version. If Python is installed, you'll see the version number.
- Install pip: If you don't have pip, install it using get-pip.py from https://bootstrap.pypa.io/get-pip.py

Creating a Django Project:

- Open a terminal: Navigate to your desired project directory.
- Create a project: Use the following command, replacing mysite with your project name:

```
django-admin startproject mysite
```

Running the Development Server:

- Navigate to the project directory: Use cd mysite.
- Start the server: Run python manage.py runserver.

Access Your Application:

Open http://127.0.0.1:8000/ in your web browser. You should see the Django welcome page, indicating a successful setup.

?**MongoDB:**

MongoDB is a flexible and scalable NoSQL database that stores data in a JSON-like format. It provides high performance, horizontal scalability, and seamless integration with Node.js, making it ideal for handling large amounts of structured and unstructured data.

Set up a MongoDB database to store your application's data.
Download: https://www.mongodb.com/try/download/community

Installation instructions: https://docs.mongodb.com/manual/installation/

?**HTML, CSS, and JavaScript**: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

?**Database Connectivity**: Use a MongoDB driver or an Object-Document Mapping (ORM) library like Djongo to connect your Django server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations.

?**Development Environment**: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

• Visual Studio Code: Download from https://code.visualstudio.com/download

To run the existing ShopEZ App project downloaded from Google Drive, Follow below steps:
Download the code from the drive link provided below.

https://drive.google.com/drive/folders/12qlo0Gy8zO16HvU2flB5seBVXXH_cqtx?usp=sharing
Then, open the project in a suitable ide or code editor and run the application using the run button provided.

• The ShopEZ app will be accessible at http://127.0.0.1:8000/

You have successfully installed and set up the ShopEZ application on your local machine. You can now proceed with further customization, development, and testing as needed.