Planning Analysis Sheet for the ShopApp Django

Project Name:

shop-app-django

Design:
For the homepage, I started by creating a simple layout featuring sliders that showcase various products. Below the sliders, there are two cards that allow users to access the filtered pages for shoes and clothes, categorized by gender.

On the shoes and clothes pages, users can view all products and apply additional filters to refine their search. Only the admin has the ability to create shoe and clothing products through the admin page. The registration functionality is provided on a separate page called "registration," which allows users to sign up using a customized form.

On this page, users can also navigate directly to the login page if they’re already registered. Additionally, users who have forgotten their passwords can reset them via email.

This functionality is set up, but I have used a placeholder for the email address for privacy reasons, indicating where the email and password should be configured to send reset codes.

The login process involves a form that prompts users for their username and password, which must match the credentials stored in the database.

Once logged in, users can access their profile page, where they can update their account details, send and receive messages from other users, and the admin can delete accounts if needed.

Development Process:

I started by creating a virtual environment with the command:
`python -m venv venv`
Then, I activated the environment using:
`venv\Scripts\activate` (for Windows).
Next, I installed Django with the command:
`pip install django`
I initiated the project with:
`django-admin startproject shopapp`
Then, I changed my directory to the project folder:
`cd shopapp`
I created the necessary applications by running:
`python manage.py startapp shoes`
, and repeated the same for the `clothes` and `users` apps.

I made sure to add each new application to the project settings as they were created.

I focused on building my applications using the Model-Template-View (MTV) methodology, starting with the model, followed by the view, then the template, and finally the URL routing to display the templates.

During development, I implemented all required functionalities, such as the inbox, message exchange, registration, login, and logout processes. Before deployment, I installed Gunicorn and Whitenoise, configured everything in `settings.py`, and collected the static files. Additionally, I created a `.gitignore` file to avoid committing sensitive files such as `.env` (used to store sensitive variables like `SECRET_KEY`) and the `venv` directory.

Credentials for Admin functionality

Username= admin
Password = admin