

Noor.AI – Skincare Chatbot Feature Development Report

1. Introduction

The Skincare Chatbot feature of Noor.AI uses OpenAI’s API integrated with a custom product database to deliver natural language recommendations and support. This chatbot provides users with answers to their skincare concerns, including product suggestions based on skin type. This feature was built independently for testing purposes before integration with the full Noor.AI web application.

2. Tools & Technologies Used

Tool	Purpose
Python	Core programming for backend logic and API interaction
OpenAI API	Natural language processing and response generation
Flask	Backend framework for chatbot API setup
HTML/CSS	Frontend design and chatbot interface
Gemini & Claude.ai	Support in writing Flask server and HTML interface
Cursor.ai	Development environment
Git & GitHub	Version control and collaboration
Kaggle Dataset	Source of product recommendations and skin-type classification

3. Dataset Usage

The same curated skincare product dataset used in the recommendation system was repurposed for the chatbot feature. The dataset contains product details, recommended skin types (dry, oily, normal), and purchase links to the official websites. The chatbot accesses this database to return the most relevant results based on user queries.

4. Feature Workflow

- User types a skincare-related query (e.g., 'What's a good moisturizer for oily skin?') in the chatbot UI.
- Flask backend receives the query and fetches relevant products from the local database.
- OpenAI's API processes the query and formats a natural language response.
- The response, including product names and links, is sent back and displayed in the UI.
- Users can click on product links to visit the official purchase page.

5. Challenges Faced

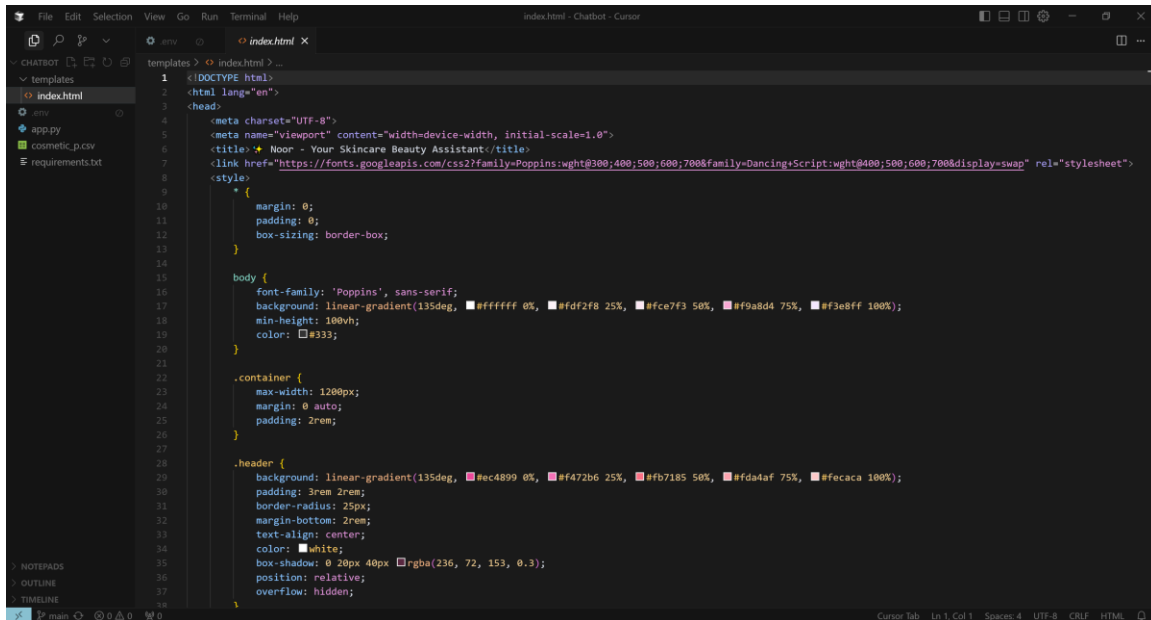
- Integrating a dynamic conversation model with a static product database.
- Ensuring natural and context-aware replies by tuning prompt templates sent to the OpenAI API.
- Handling ambiguity in user queries and mapping them to available product categories.

6. Future Improvements

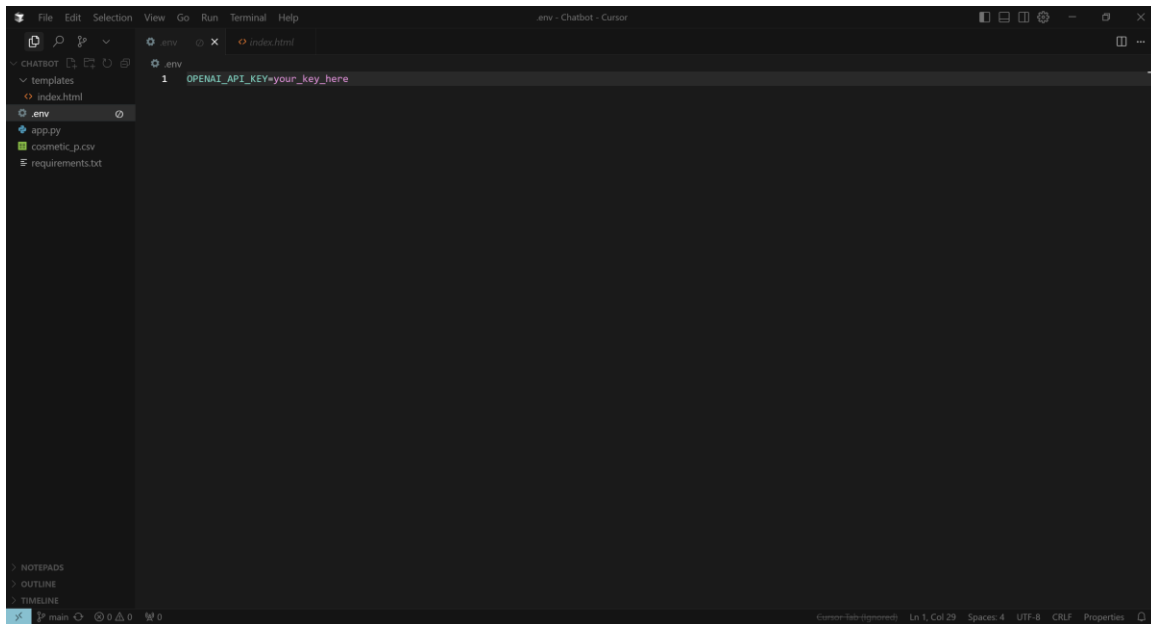
- Introduce user authentication so the chatbot can give personalized recommendations based on skin history.
- Add a rating system to learn from user feedback and improve suggestions.
- Integrate a real-time search to expand database coverage without manual updates.

7. Screenshots

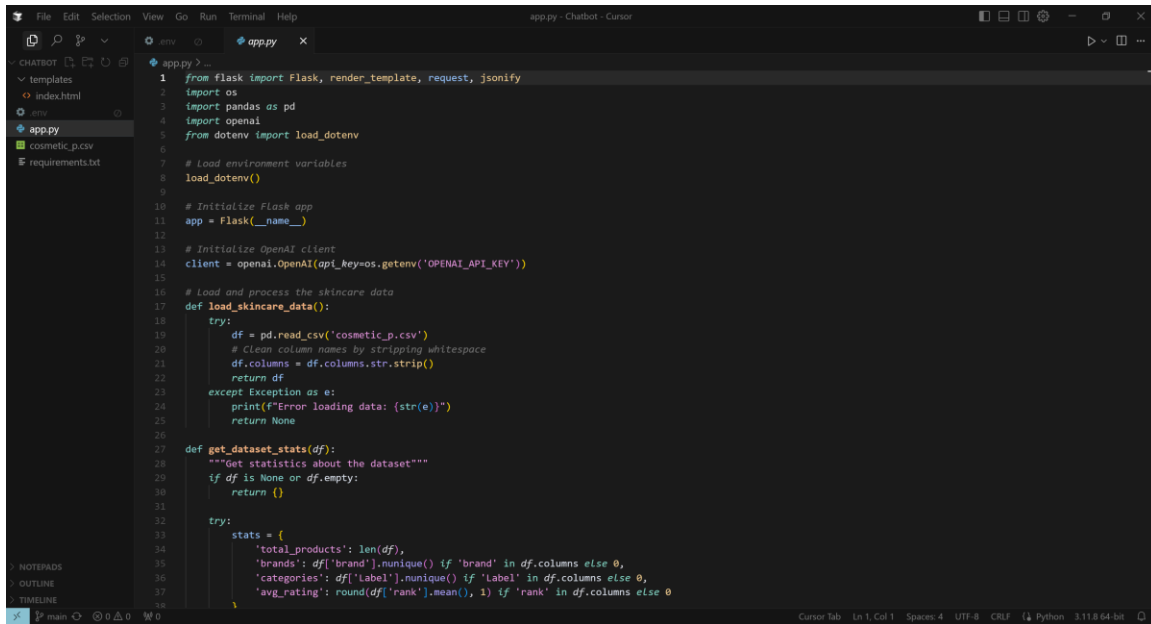
Screenshots showcasing the chatbot interface and responses will be added after full UI integration.



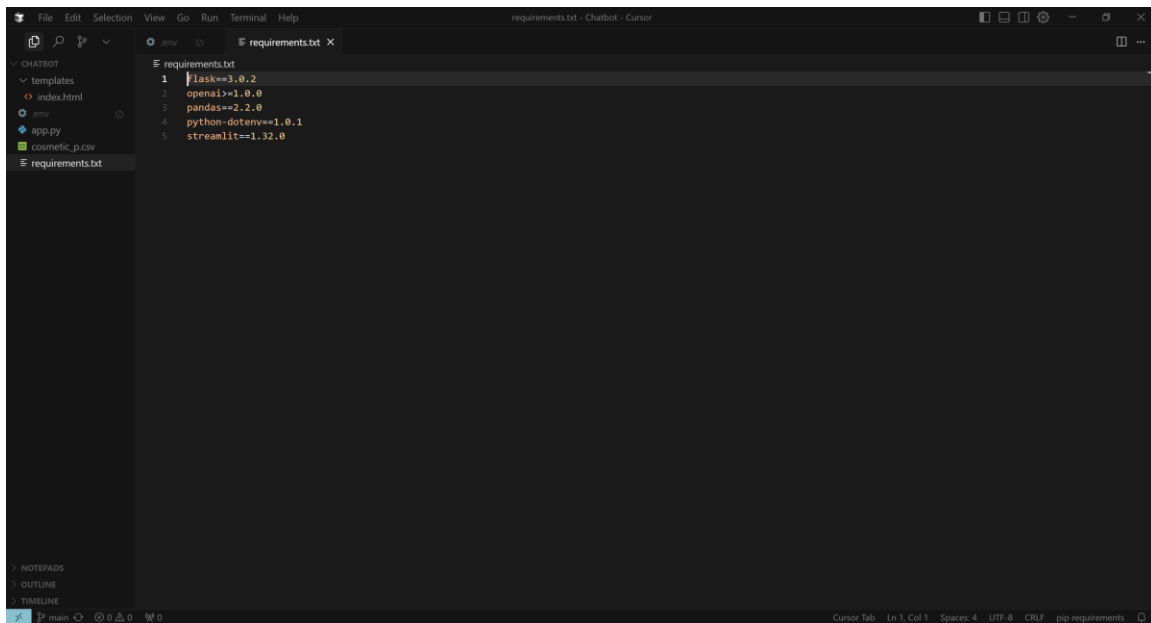
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title> Noor - Your Skincare Beauty Assistant</title>
7   <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;500;600;700&family=Dancing+Script:wght@400;500;600;700&display=swap" rel="stylesheet">
8   <style>
9     * {
10       margin: 0;
11       padding: 0;
12       box-sizing: border-box;
13     }
14
15     body {
16       font-family: 'Poppins', sans-serif;
17       background: linear-gradient(135deg, #ffffff 0%, #Fdf2f8 25%, #fce7f3 50%, #f9a8d4 75%, #f3e8ff 100%);
18       min-height: 100vh;
19       color: #333;
20     }
21
22     .container {
23       max-width: 1200px;
24       margin: 0 auto;
25       padding: 2rem;
26     }
27
28     .header {
29       background: linear-gradient(135deg, #ec4899 0%, #f472b6 25%, #fb7185 50%, #fda4af 75%, #fecaca 100%);
30       padding: 3rem 2rem;
31       border-radius: 25px;
32       margin-bottom: 2rem;
33       text-align: center;
34       color: white;
35       box-shadow: 0 20px 40px rgba(236, 72, 153, 0.3);
36       position: relative;
37       overflow: hidden;
38     }
```



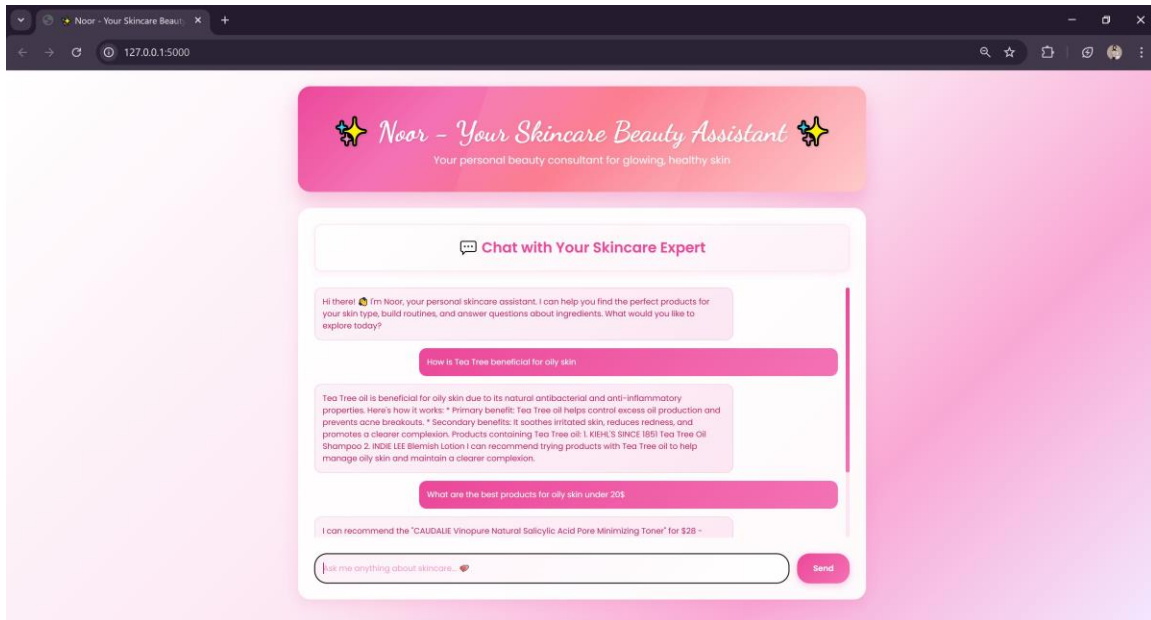
```
1 OPENAI_API_KEY=your_key_here
```



```
1 from flask import Flask, render_template, request, jsonify
2 import os
3 import pandas as pd
4 import openai
5 from dotenv import load_dotenv
6
7 # Load environment variables
8 load_dotenv()
9
10 # Initialize Flask app
11 app = Flask(__name__)
12
13 # Initialize OpenAI client
14 client = openai.OpenAI(api_key=os.getenv('OPENAI_API_KEY'))
15
16 # Load and process the skincare data
17 def load_skincare_data():
18     try:
19         df = pd.read_csv('cosmetic_p.csv')
20         # Clean column names by stripping whitespace
21         df.columns = df.columns.str.strip()
22         return df
23     except Exception as e:
24         print(f"Error loading data: {str(e)}")
25         return None
26
27 def get_dataset_stats(df):
28     """Get statistics about the dataset"""
29     if df is None or df.empty:
30         return {}
31
32     try:
33         stats = {
34             'total_products': len(df),
35             'brands': df['brand'].nunique() if 'brand' in df.columns else 0,
36             'categories': df['label'].nunique() if 'label' in df.columns else 0,
37             'avg_rating': round(df['rank'].mean(), 1) if 'rank' in df.columns else 0
38         }
```



```
1 flask==3.0.2
2 openai==1.0.0
3 pandas==2.2.0
4 python-dotenv==1.0.1
5 streamlit==1.32.0
```



8. Conclusion

The AI-powered skincare chatbot successfully demonstrates how conversational AI can enhance user interaction and guidance in skincare product selection. This module is tested and ready for integration with the Noor.AI web platform.