I'd be glad to assist you with creating Python Django code that incorporates a prompt-based product recommendation system using LangChain. Here's a comprehensive approach that leverages LangChain's capabilities for natural language processing and product search:

1. Project Setup:

- Create a new Django project named product_recommender:

Bash

```bash
django-admin startproject product_recommender
```

- Navigate to the project directory:

Bash

```bash
cd product_recommender
```

- Create a Django app named recommendations to house your recommendation logic:

Bash

```bash
python manage.py startapp recommendations
```

2. Install LangChain:

- Install LangChain using pip:

Bash

```bash
pip install langchain
```

3. Model Definition (models.py):

- In the recommendations/models.py file, define a model to represent your products:

Python

```
from django.db import models

class Product(models.Model):
    name = models.CharField(max_length=255)
    description = models.TextField(blank=True)
    # Add other product-related fields as needed
```

4. LangChain Integration (langchain_utils.py):

- Create a new file named langchain_utils.py within the recommendations app directory to handle LangChain interactions:

Python

```
from langchain.llms import AutoLMS
from langchain.prompts import Chain, Task

# Replace with your preferred large language model (LLM)
model_name = "facebook/bart-base"  # Example LLM

def get_product_recommendations(prompt):
    # Initialize the LLM
    llm = AutoLMS.from_pretrained(model_name)

    # Define the LangChain prompt structure
    chain = Chain(
        tasks=[
            Task(
                inputs=prompt,
                llm_outputs=["top_3_product_recommendations"],
            )
        ]
    )

    # Generate recommendations using LangChain
    outputs = llm.run(chain)
    recommendations = outputs[0]["top_3_product_recommendations"]

    # Process and return recommendations (replace with your logic)
    return recommendations[:3]  # Return top 3 recommendations
```

5. View for Handling User Input (views.py):

- In the recommendations/views.py file, create a view to handle user input and generate recommendations:

Python

```python
from django.shortcuts import render
from .langchain_utils import get_product_recommendations

def recommend_products(request):
    if request.method == 'POST':
        prompt = request.POST.get('prompt')
        recommendations = get_product_recommendations(prompt)
        context = {'recommendations': recommendations}
    else:
        context = {}
    return render(request, 'recommendations/recommend_products.html',
context)
```

6. Template for Displaying Recommendations (recommend_products.html):

- Create a template named recommend_products.html within the recommendations app's templates directory:

HTML

```html
<!DOCTYPE html>
<html>
<head>
    <title>Product Recommender</title>
</head>
<body>
    <h1>Product Recommendations</h1>
    <form method="POST">
        <label for="prompt">Enter a prompt:</label>
        <textarea id="prompt" name="prompt"></textarea>
        <button type="submit">Recommend Products</button>
    </form>
    {% if recommendations %}
```

```
        <h2>Recommendations</h2>
        <ul>
            {% for recommendation in recommendations %}
                <li>{{ recommendation }}</li>
            {% endfor %}
        </ul>
    {% endif %}
</body>
</html>
```

7. URL Configuration (urls.py):

- Update the recommendations/urls.py file to map the view to a URL:

Python

```python
from django.urls import path
from . import views

urlpatterns = [
    path('', views.recommend_products, name='recommend_products'),
]
```

8. Project URL Configuration (product_recommender/urls.py):

- In the main project's product_recommender/urls.py file, include the app's URLs:

Python

```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('recommendations/', include('recommendations.urls')),
]
```

9. Running the Application:

- Apply database migrations:

Sources

1. https://github.com/Pashok-zh/store
2.

https://dev.to/cindyachieng/laughing-blog-tutorial-series-part-2--creating-a-newsletter-app-3dph
3. https://github.com/Nguyeenvuu/movielens_api