**AI-Powered Veterinary Appointment Scheduler – Documentation**

**Overview**

The AI-Powered Veterinary Appointment Scheduler is designed to **automate the process of filling canceled appointment slots** in a veterinary clinic. Using **LangChain** and **Hugging Face’s flan-t5-base model**, the system evaluates the waitlist and selects the best candidate based on urgency, client flexibility, and fairness.

This documentation explains the **architecture, components, usage, and customization options**.

**Architecture**

The system follows a **prompt-based AI workflow**:

1. **Input:** A canceled appointment with details of the service type, time slot, and current waitlist.
2. **Prompt Generation:** Uses a **LangChain PromptTemplate** to format the input for the AI model.
3. **AI Model:** Calls **Hugging Face flan-t5-base** via HuggingFaceHub to analyze the prompt and generate a recommendation.
4. **Output:** Returns the **selected client and reasoning**, which can be displayed or used for automated scheduling notifications.

**Flow Diagram:**

Canceled Appointment + Waitlist --> PromptTemplate --> HuggingFaceHub LLM --> Output: Recommended Client

**Components**

**1. LangChain**

* PromptTemplate: Creates structured prompts from appointment and waitlist data.
* LLMChain: Connects the prompt template with the LLM to generate AI responses.

**2. HuggingFaceHub**

* Provides access to google/flan-t5-base.
* Parameters:
  + temperature: Controls creativity of responses.
  + max\_length: Limits the output token length.

**3. Input Data**

* **service\_type**: Type of appointment (e.g., vaccination, surgery).
* **time\_slot**: Date and time of the canceled appointment.
* **waitlist**: List of clients waiting for appointments, including urgency, pet type, and flexibility.

**Setup & Installation**

**Prerequisites**

* Python 3.10+
* Hugging Face API token with **read access** to google/flan-t5-base
* Install required packages:

pip install langchain huggingface-hub

**Configuration**

import os

os.environ["HUGGINGFACEHUB\_API\_TOKEN"] = "your\_huggingface\_token\_here"

**Usage**

**Initialize LLM**

from langchain import PromptTemplate, LLMChain

from langchain.llms import HuggingFaceHub

llm = HuggingFaceHub(

repo\_id="google/flan-t5-base",

model\_kwargs={"temperature": 0.6, "max\_length": 200}

)

**Define Prompt Template**

template = """

You are an AI scheduling assistant for a veterinary clinic.

A client canceled their appointment for {service\_type} at {time\_slot}.

Here is the waitlist:

{waitlist}

Select the best replacement client considering urgency, flexibility, and fairness.

Output format:

Client: <name>

Reason: <short explanation>

"""

prompt = PromptTemplate(

input\_variables=["service\_type", "time\_slot", "waitlist"],

template=template

)

**Combine Prompt + LLM**

chain = LLMChain(llm=llm, prompt=prompt)

**Run Example**

service\_type = "vaccination"

time\_slot = "Monday 10:00 AM"

waitlist = """

1. Sarah - vaccination for puppy, flexible any time

2. Tom - surgery follow-up, needs early slot

3. Alice - routine check-up, prefers afternoons

"""

response = chain.run(service\_type=service\_type, time\_slot=time\_slot, waitlist=waitlist)

print(response)

**Future Enhancements**

* Automatic notifications via email or SMS to selected clients.
* Multi-slot auto-fill for multiple cancellations at once.
* Support for multiple AI models for faster or localized recommendations.
* Integration with frontend dashboards for real-time scheduling updates.