**Ollama** is a platform designed for running and interacting with large language models (LLMs) on local devices. It allows users to download, install, and run LLMs on their own hardware without needing to rely on cloud services. This can improve privacy, reduce costs, and offer better performance, particularly for users with strong local computational resources.

**Key Features of Ollama:**

1. **Local Model Running**: Users can run sophisticated AI language models locally, allowing for data security and privacy, as the data doesn’t need to be sent to external servers.
2. **Model Variety**: Ollama provides access to a variety of LLMs, such as those tailored for code generation, natural language processing, chat, and more.
3. **Cost-Effective**: Since the models run locally, users do not incur costs associated with using cloud-based AI models, which typically charge for every inference or usage.
4. **Performance**: By leveraging local hardware, particularly powerful CPUs and GPUs, users can achieve faster model responses and avoid latency issues associated with internet-based services.
5. **Customizable and Extendable**: Ollama allows for the customization and extension of models to fit specific user needs, giving flexibility in how the LLMs can be used.

**Use Cases:**

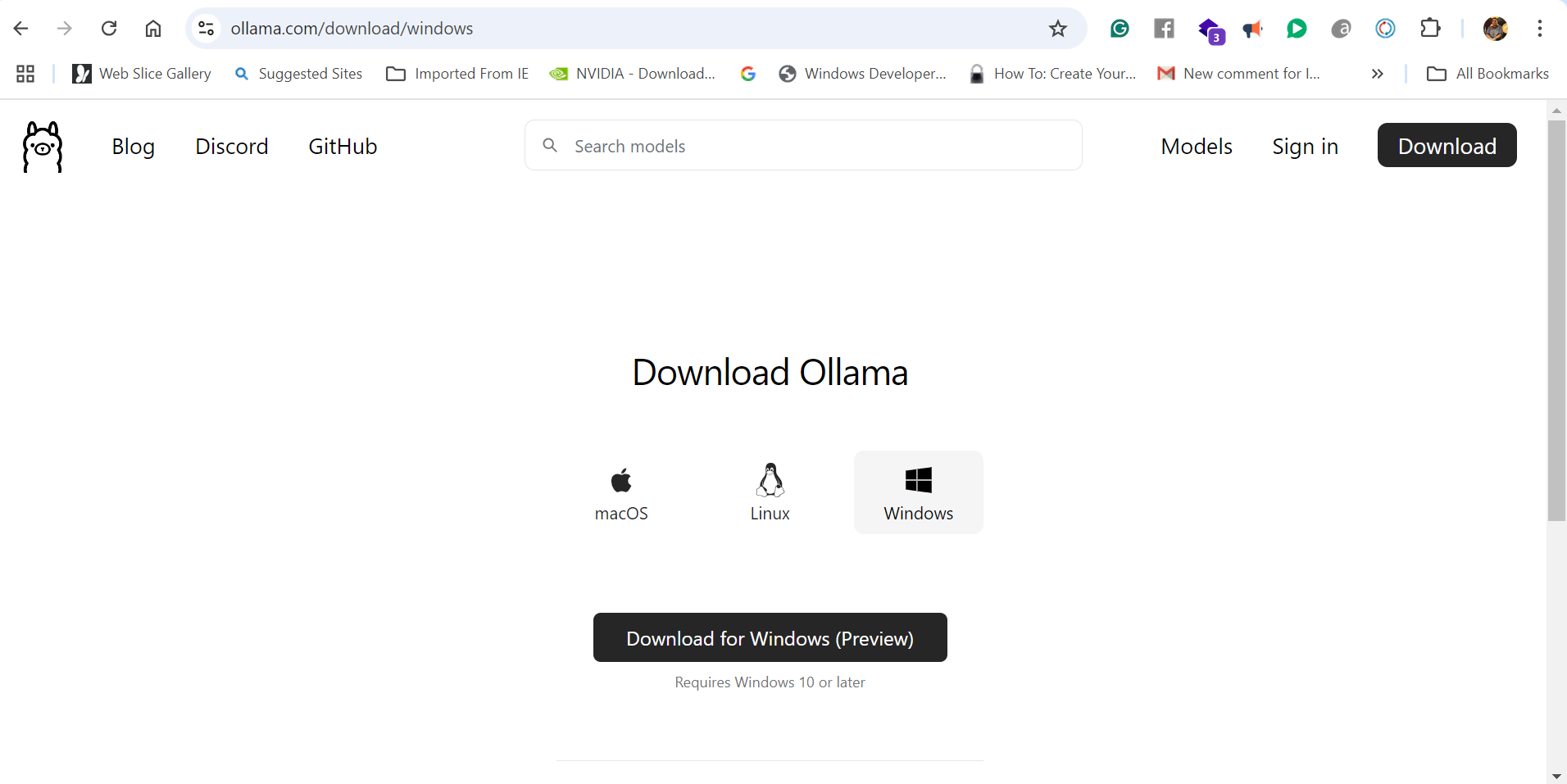
* Developers who want to experiment with AI models locally.
* Organizations requiring high levels of privacy, where data cannot be shared with external servers.
* Users who want to avoid cloud costs and latency associated with cloud-based LLMs.
* Individuals looking to integrate LLMs into local applications without requiring continuous internet access.

In essence, Ollama provides a self-hosted, local-first approach to using LLMs, offering an alternative to cloud-based AI platforms like OpenAI or Google Cloud AI.

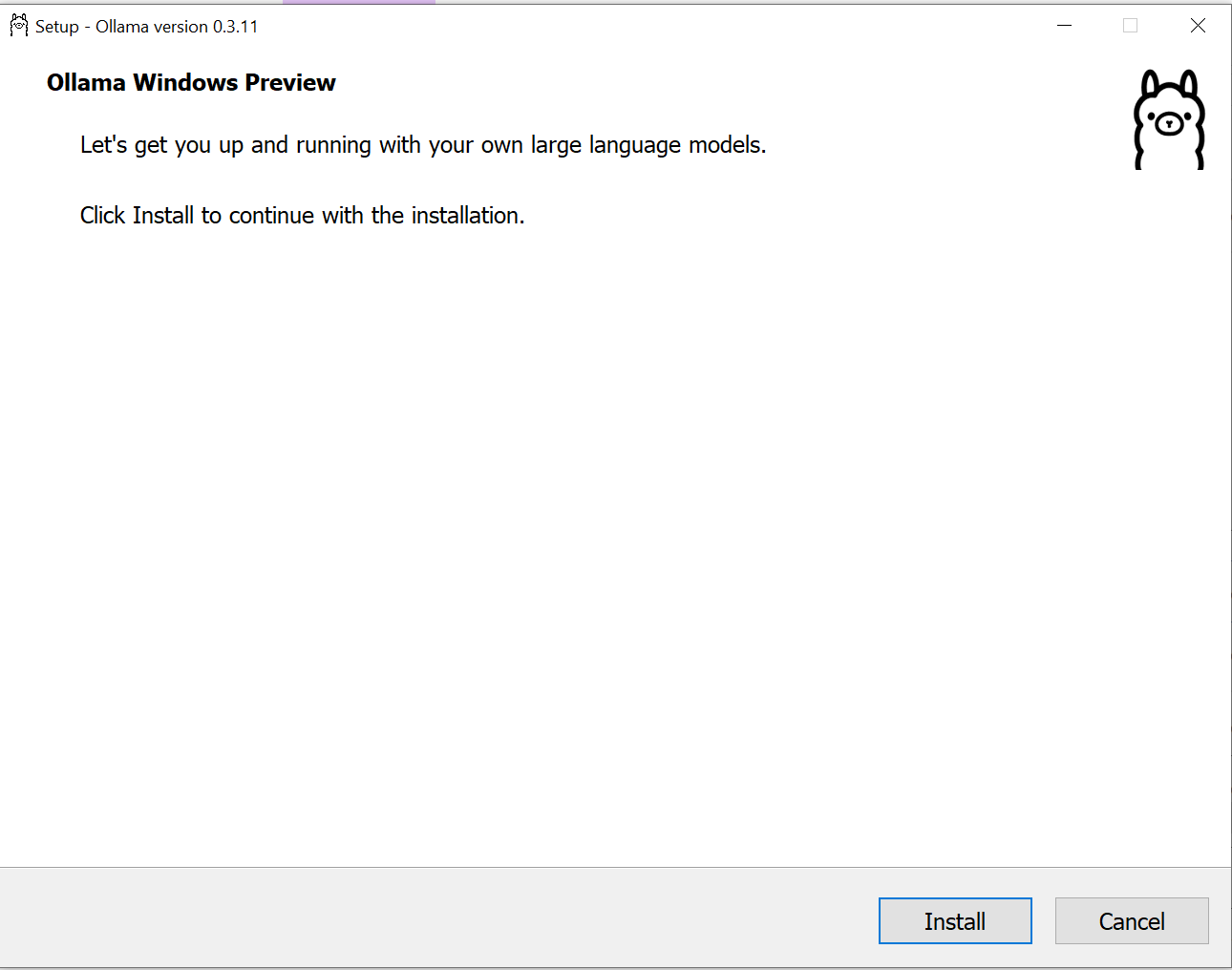
To get started with ollama we need to download it

Here is the link.

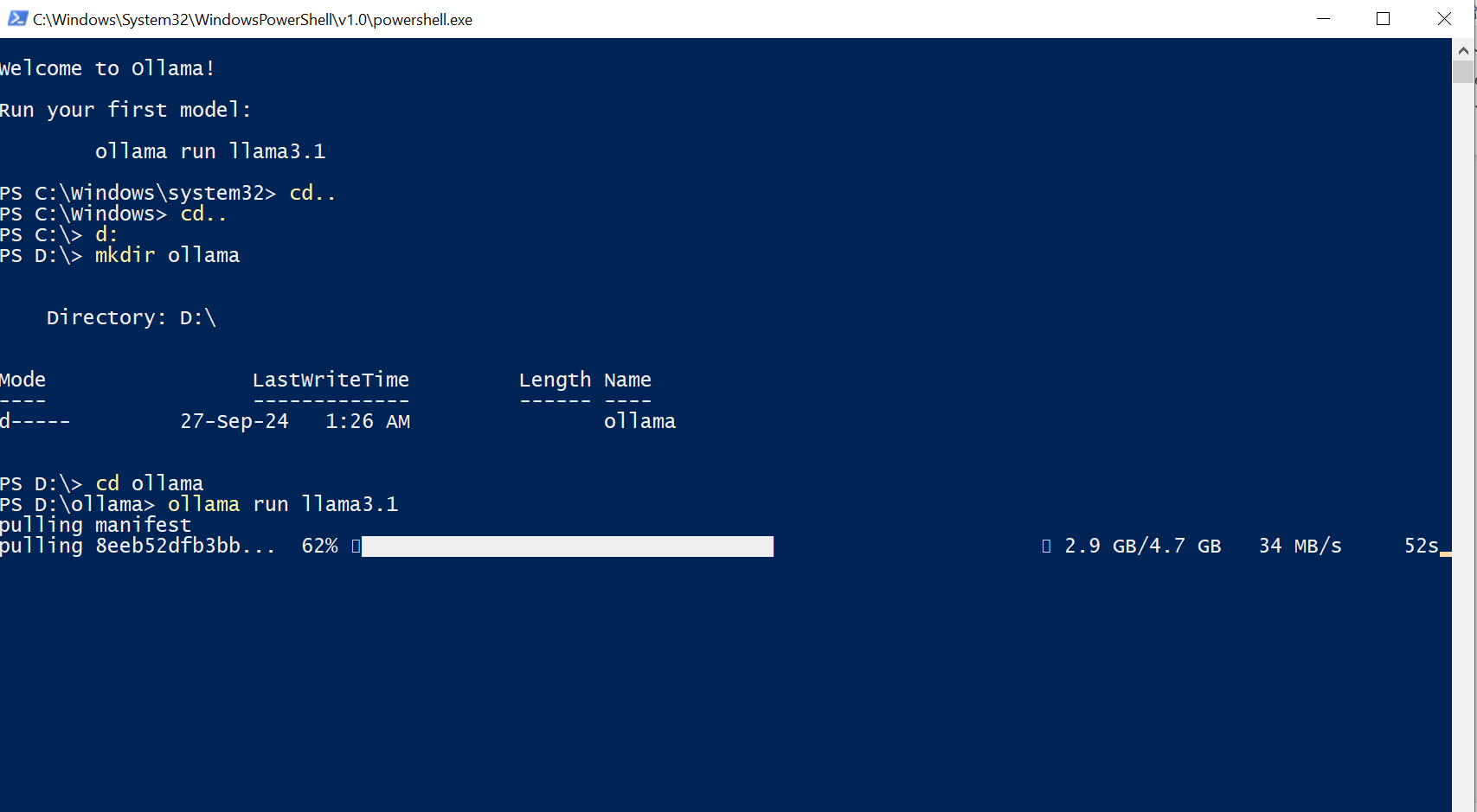
<https://ollama.com/download/windows>



Install the .exe file that gets downloaded



After installation downloading Ollama 3.1



Let’s create a demo with Ollama

Make sure Ollama is running.

We enable the Anaconda environment.

Now on VScode write the following code.

import os

from dotenv import load\_dotenv

from langchain\_community.llms import Ollama

import streamlit as st

from langchain\_core.prompts import ChatPromptTemplate

from langchain\_core.output\_parsers import StrOutputParser

load\_dotenv()

## Langsmith Tracking

#os.environ["LANGCHAIN\_API\_KEY"]=os.getenv("LANGCHAIN\_API\_KEY")

#os.environ["LANGCHAIN\_TRACING\_V2"]="true"

#os.environ["LANGCHAIN\_PROJECT"]=os.getenv("LANGCHAIN\_PROJECT")

## Prompt Template

prompt=ChatPromptTemplate.from\_messages(

    [

        ("system","You are a helpful assistant. Please respond to the question asked"),

        ("user","Question:{question}")

    ]

)

## streamlit framework

st.title("Langchain Demo With Llama 3.1 Model")

input\_text=st.text\_input("What question you have in mind?")

## Ollama Llama2 model

llm=Ollama(model="llama3.1")

output\_parser=StrOutputParser()

chain=prompt|llm|output\_parser

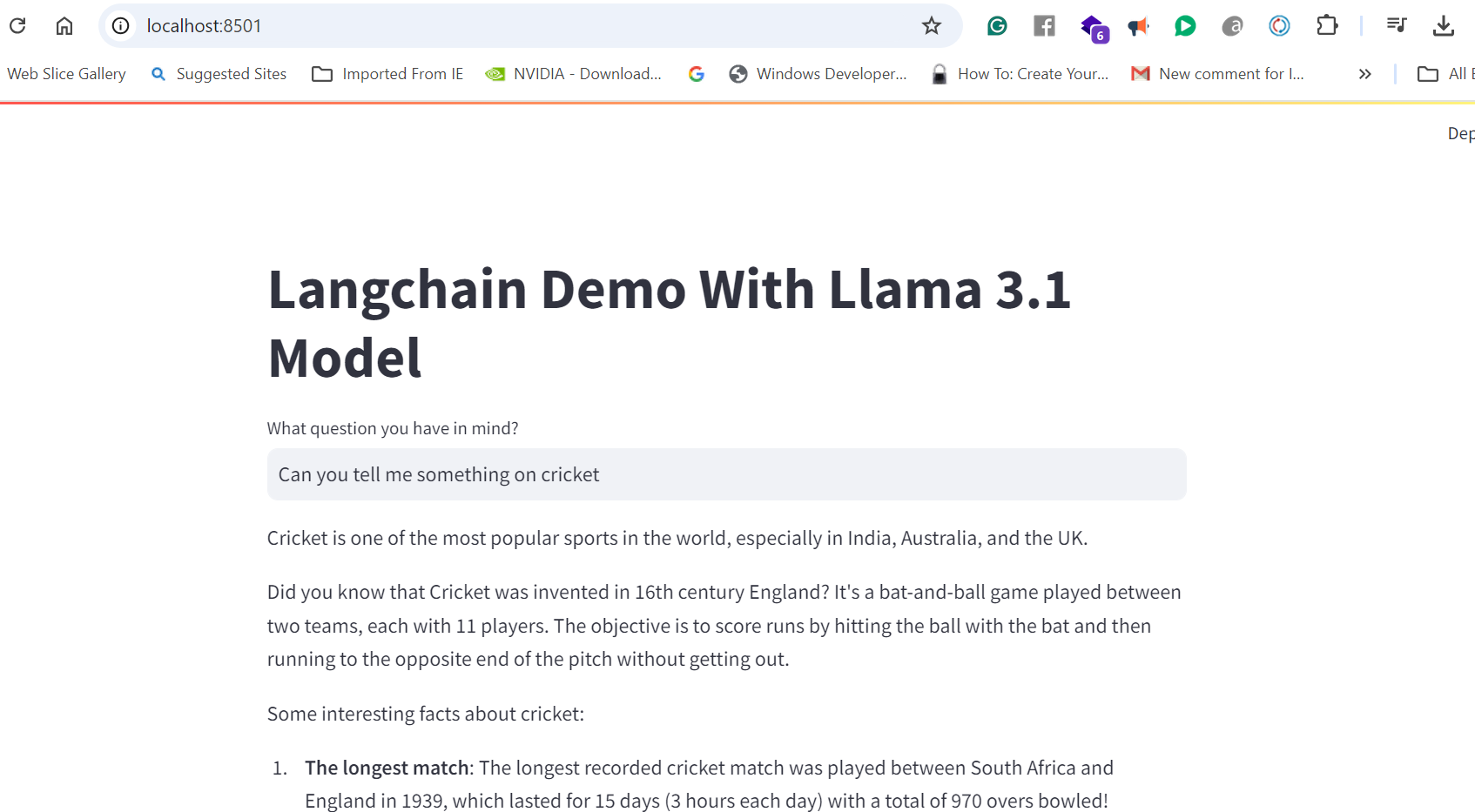
if input\_text:

    st.write(chain.invoke({"question":input\_text}))

run the code in anaconda environment and use the following command.

Streamlit run app.py

The app will open up in the default web browser and we can run a query depending on the PC’s cpu you will get faster inference.



You are ready with your first Llama app running on CPU.