1. 前期准备

本文档目的是帮助在本地linux服务器端，部署并成功运行Meta开源LLM llama2。基于本文档已经成功在A100 和8卡4090服务器部署llama2-7b-chat-hf。

开始本文档，需要提前准备：

1. 支撑llama2运行的GPU服务器资源，使用基于Linux内核的服务器操作系统。
2. 确保可以通过远程连接，连接到服务器资源。（比如使用vscode remote-ssh)
3. 确保有足够的存储空间，最好有50G的富裕空间。
4. 服务器已经预装python3, CUDA等基础软件，由于本教程使用docker部署，所以不需要安装miniconda。

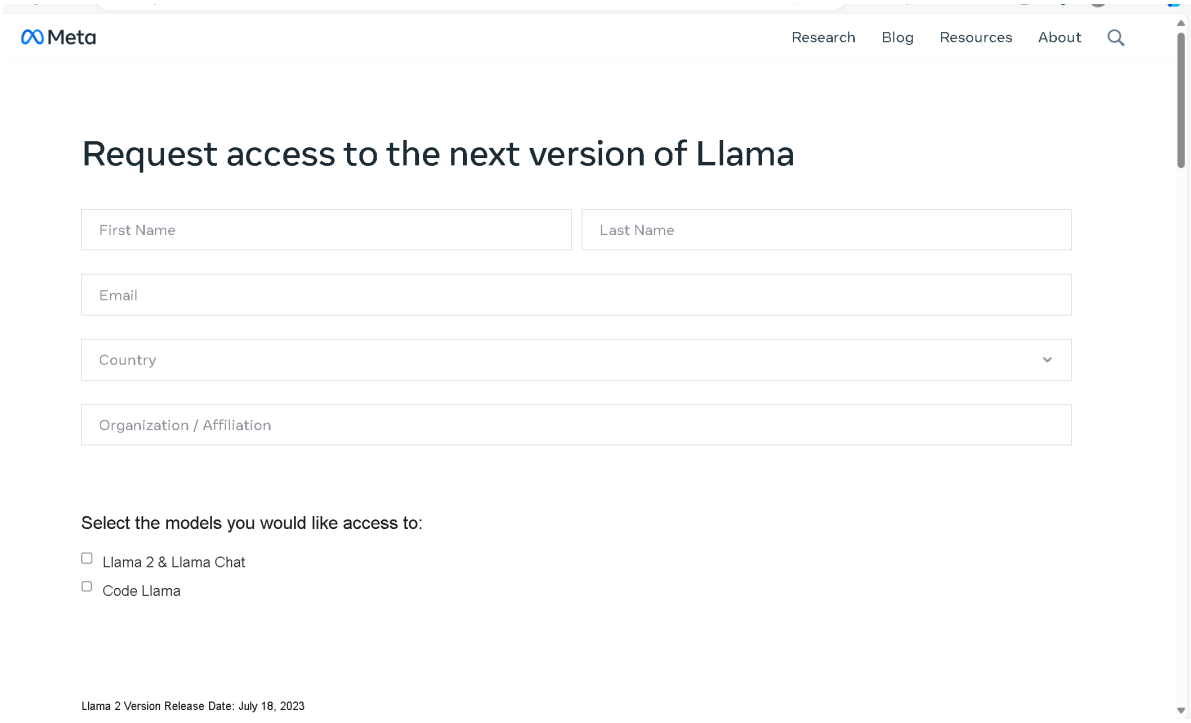
2. 下载llama-2

llama2可以在Meta官网和Huggingface网站上获取，为了安装的便捷性。我们最好遵从以下工作流完成：

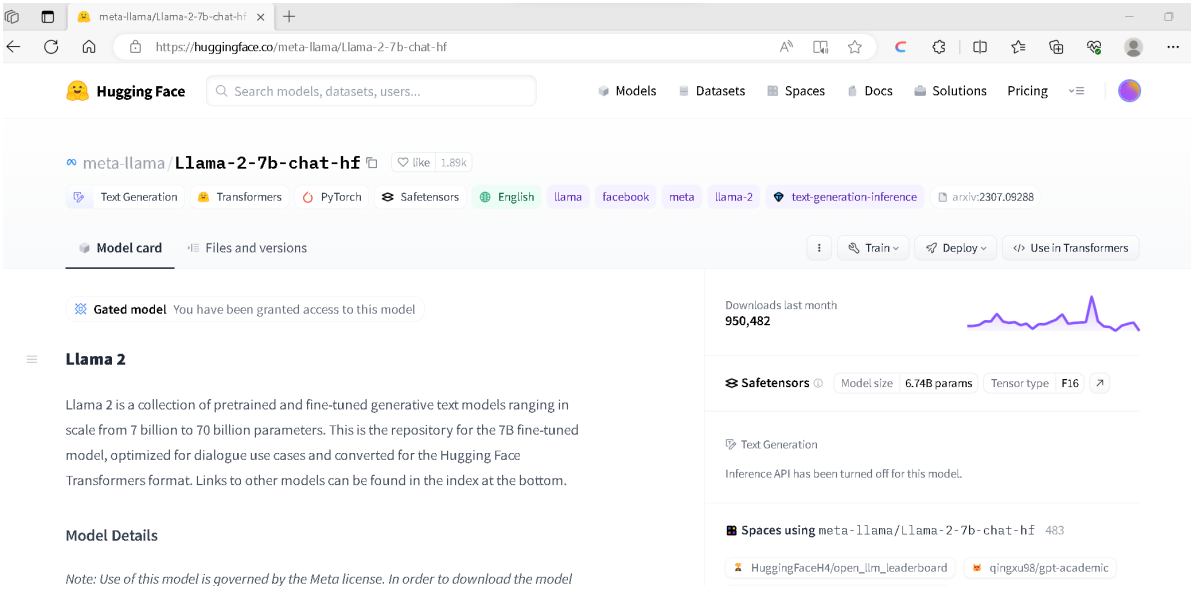
1. Meta官网，使用美国节点+Gmail注册，并申请使用llama2的权限。
2. Huggingface官网，使用1中的gmail注册，并在Huggingface界面再次申请llama2权限。
3. 两个权限都申请通过以后，就可以开始准备下载环节。

注意：这里的两次注册最好使用同一个邮箱，且确保自己在美国节点的代理条件之下，预计两次审核时间不会超过一个小时。

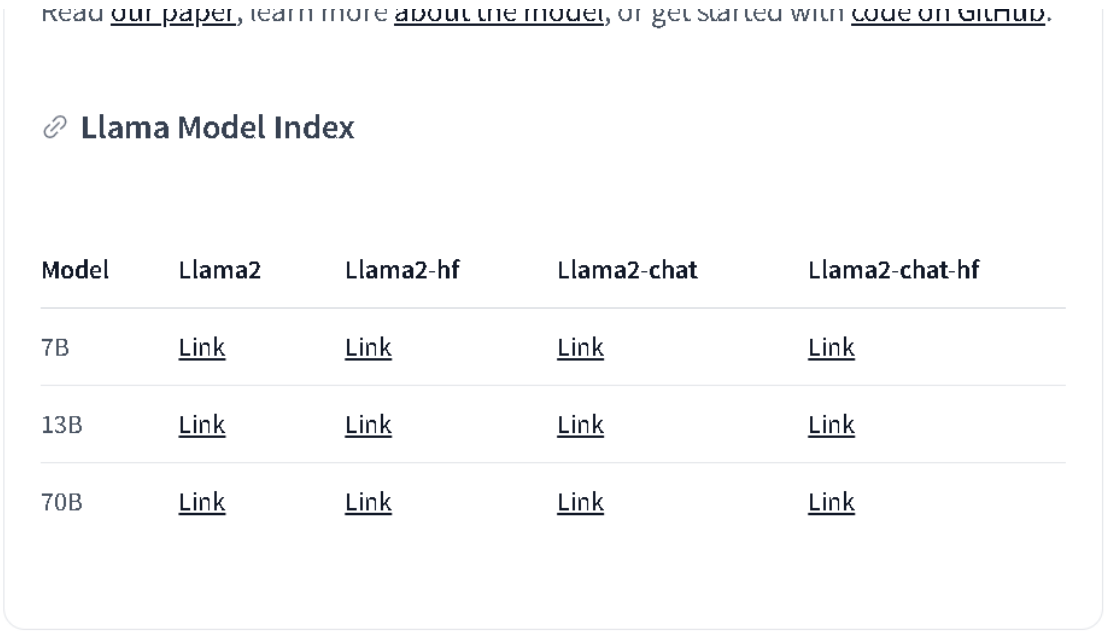
Meta官网在完成如下表格的申请后，将会发送反馈邮件：



huggingface在申请完成后，将会出现以下界面：



在申请完成之后，我们就可以下载Huggingface的llama2版本。



Read our paper, learn more about the model, or get started with code on GitHub.

Llama Model Index

Model	Llama2	Llama2-hf	Llama2-chat	Llama2-chat-hf
7B	Link	Link	Link	Link
13B	Link	Link	Link	Link
70B	Link	Link	Link	Link

llama2有许多版本，如上图，7B 13B是参数规模，不确定计算资源能力的情况下，先选择7B版本，实测3卡4090可以轻松运行7B 版本。

后缀主要是功能和版本的不同，如本文档选择的llama2-7b-chat-hf便是7B参数规模的对话机器人，自带使用Gradio构建的简易交互界面。

考虑到网络问题，我们需要使用的是Huggingface的镜像站和Hugg官方下载工具`huggingface-cli`

镜像站在：[hf-mirror.com - Huggingface 镜像站]

按照以下方式执行，可以得到下载在当前目录下的具体文件。

```
pip install -U huggingface_hub # 安装

export HF_ENDPOINT=https://hf-mirror.com #切换镜像源，每次使用huggingface-cli镜像源下载都需要提前执行这一条命令

huggingface-cli download  --resume-download meta-llama/Llama-2-7b-chat-hf  --local-dir Llama-2-7b-chat-hf  --local-dir-use-symlinks False  --token hf_**Your token** # 替换自己的Huggingface token后，即可下载到本地，注意参数--local-dir-use-symlinks 必须添加，不然大于5M的文件会被自动生成软链接，后续llama2无法启动


```

这样我们就下载完成了llama2-7b-chat-hf。

3. Docker运行

本文档运行环境为docker,优点是环境隔离足够强，且由于docker对端口需要映射，可以解决后续一个很麻烦的端口问题。

（目前开源模型，llama2和stable diffusion等等，开源仓库都使用7860端口作为交互端口，如果服务器7860端口被占用，就需要腾出端口或者修改工程文件来实现。但是修改工程文件需要花费更多时间寻找对应工程文件，docker 的端口映射可以帮助我们很简单地解决这个问题。）

要在docker环境下运行，自然需要安装docker,具体可以参考参考链接一节。自己配置docker file和环境依赖很耗时间，本着不重复造轮子的原则，使用前辈已经配置好的docker 库文件来完成。

库文件地址：soulteary/docker-llama2-chat: Play LLaMA2 (official / 中文版 / INT4 / llama2.cpp) Together! ONLY 3 STEPS! (non GPU / 5GB vRAM / 8~14GB vRAM) (github.com)

该库已经有安装说明，可在参考链接中查阅。由于本文档定位为解决从0到1的问题，所以对该库文件的文件不作修改，力求最快速度跑通。

下载github 仓库，需要使用镜像站，可以使用kkgithub.com来下载。由于镜像站稳定性差，建议搜索需要下载时可以使用的镜像站，并参考对应命令来下载。也可以下载到本地后，使用vscode 配置SFTP上传文件。

下面我们需要完成一系列调整工作：

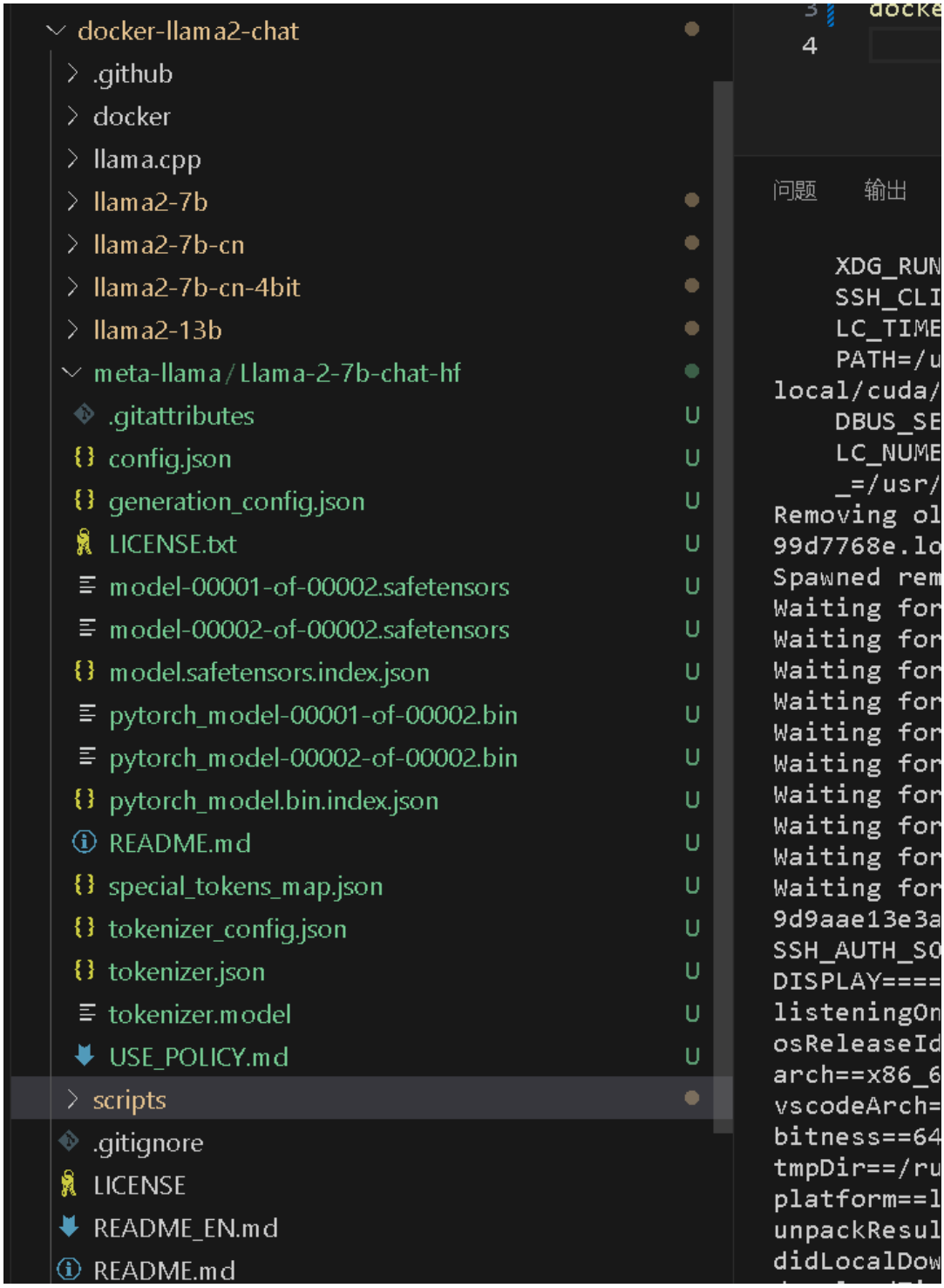
1. 调整目录结构

```
# 创建一个新的目录，用于存放我们的模型
mkdir meta-llama

# 将下载好的模型移动到目录中
mv Llama-2-7b-chat-hf meta-llama/
mv Llama-2-13b-chat-hf meta-llama/
mv Llama-2-70b-chat-hf meta-llama/


```

2. 完整的目录结构类似下面这样，所有的模型都在我们创建的 `meta-llama` 目录的下一级中

```
# tree -L 2 meta-llama
meta-llama
├── Llama-2-13b-chat-hf
|   ├── added_tokens.json
```

```
|   ├── config.json
|   ├── generation_config.json
|   ├── LICENSE.txt
|   ├── model-00001-of-00003.safetensors
|   ├── model-00002-of-00003.safetensors
|   ├── model-00003-of-00003.safetensors
|   ├── model.safetensors.index.json
|   ├── pytorch_model-00001-of-00003.bin
|   ├── pytorch_model-00002-of-00003.bin
|   ├── pytorch_model-00003-of-00003.bin
|   ├── pytorch_model.bin.index.json
|   ├── README.md
|   ├── Responsible-Use-Guide.pdf
|   ├── special_tokens_map.json
|   ├── tokenizer_config.json
|   ├── tokenizer.model
|   └── USE_POLICY.md
└── Llama-2-7b-chat-hf
    ├── added_tokens.json
    ├── config.json
    ├── generation_config.json
    ├── LICENSE.txt
    ├── model-00001-of-00002.safetensors
    ├── model-00002-of-00002.safetensors
    ├── model.safetensors.index.json
    ├── models--meta-llama--Llama-2-7b-chat-hf
    ├── pytorch_model-00001-of-00003.bin
    ├── pytorch_model-00002-of-00003.bin
    ├── pytorch_model-00003-of-00003.bin
    ├── pytorch_model.bin.index.json
    ├── README.md
    ├── special_tokens_map.json
    ├── tokenizer_config.json
    ├── tokenizer.json
    ├── tokenizer.model
    └── USE_POLICY.md

```



开始构建并运行：

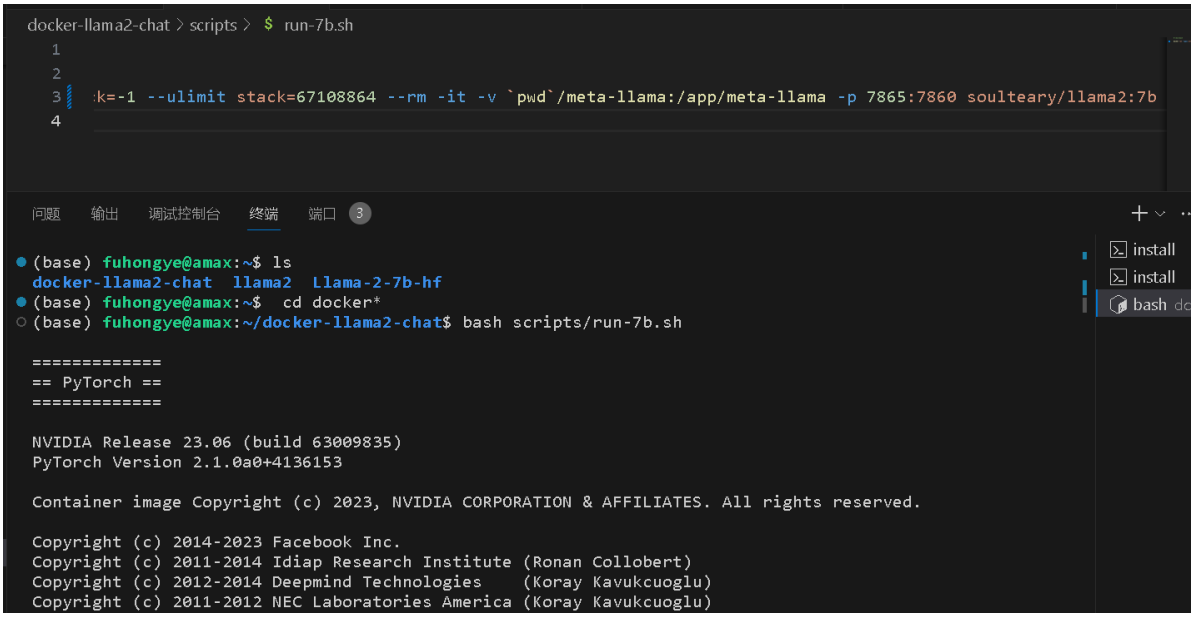
```
# 进入程序目录
cd docker-llama2-chat
# 构建 7B 镜像
bash scripts/make-7b.sh
# 或者，构建 13B 镜像
bash scripts/make-13b.sh

#构建完成后

# 运行 7B 镜像，应用程序
bash scripts/run-7b.sh
# 或者，运行 13B 镜像，应用程序
bash scripts/run-13b.sh

```

```
#命令执行后，如果一切顺利，你将看到类似下面的日志：

=============
== PyTorch ==
=============

NVIDIA Release 23.06 (build 63009835)
PyTorch Version 2.1.0a0+4136153

Container image Copyright (c) 2023, NVIDIA CORPORATION & AFFILIATES. All rights reserved.

Copyright (c) 2014-2023 Facebook Inc.
Copyright (c) 2011-2014 Idiap Research Institute (Ronan Collobert)
Copyright (c) 2012-2014 Deepmind Technologies    (Koray Kavukcuoglu)
Copyright (c) 2011-2012 NEC Laboratories America (Koray Kavukcuoglu)
Copyright (c) 2011-2013 NYU                      (Clement Farabet)
Copyright (c) 2006-2010 NEC Laboratories America (Ronan Collobert, Leon Bottou, Iain Melvin, Jason Weston)
Copyright (c) 2006      Idiap Research Institute (Samy Bengio)
Copyright (c) 2001-2004 Idiap Research Institute (Ronan Collobert, Samy Bengio, Johnny Mariethoz)
Copyright (c) 2015      Google Inc.
Copyright (c) 2015      Yangqing Jia
Copyright (c) 2013-2016 The Caffe contributors
All rights reserved.

Various files include modifications (c) NVIDIA CORPORATION & AFFILIATES.  All rights reserved.

This container image and its contents are governed by the NVIDIA Deep Learning Container License.
By pulling and using the container, you accept the terms and conditions of this license:
https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license

WARNING: CUDA Minor Version Compatibility mode ENABLED.
  Using driver version 525.105.17 which has support for CUDA 12.0.  This container
  was built with CUDA 12.1 and will be run in Minor Version Compatibility mode.
  CUDA Forward Compatibility is preferred over Minor Version Compatibility for use
  with this container but was unavailable:
  [[Forward compatibility was attempted on non supported HW (CUDA_ERROR_COMPAT_NOT_SUPPORTED_ON_DEVICE) cuInit()=804]]
  See https://docs.nvidia.com/deploy/cuda-compatibility/ for details.

Loading checkpoint shards: 100%|████████████████████████████████████████████████████████████████████████████████████████████████████████| 2/2 [00:05<00:00,  2.52s/it]
Caching examples at: '/app/gradio_cached_examples/20'
Caching example 1/5
```

```
/usr/local/lib/python3.10/dist-packages/transformers/generation/utils.py:1270: UserWarning: You have modified the pretrained model configuration to control generation. This is a deprecated strategy to control generation and will be removed soon, in a future version. Please use a generation configuration file (see https://huggingface.co/docs/transformers/main_classes/text_generation )
  warnings.warn(
Caching example 2/5
Caching example 3/5
Caching example 4/5
Caching example 5/5
Caching complete

/usr/local/lib/python3.10/dist-packages/gradio/utils.py:839: UserWarning: Expected 7 arguments for function <function generate at 0x7f3e096a1000>, received 6.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/gradio/utils.py:843: UserWarning: Expected at least 7 arguments for function <function generate at 0x7f3e096a1000>, received 6.
  warnings.warn(
Running on local URL:  http://0.0.0.0:7860

To create a public link, set `share=True` in `launch()`.
```

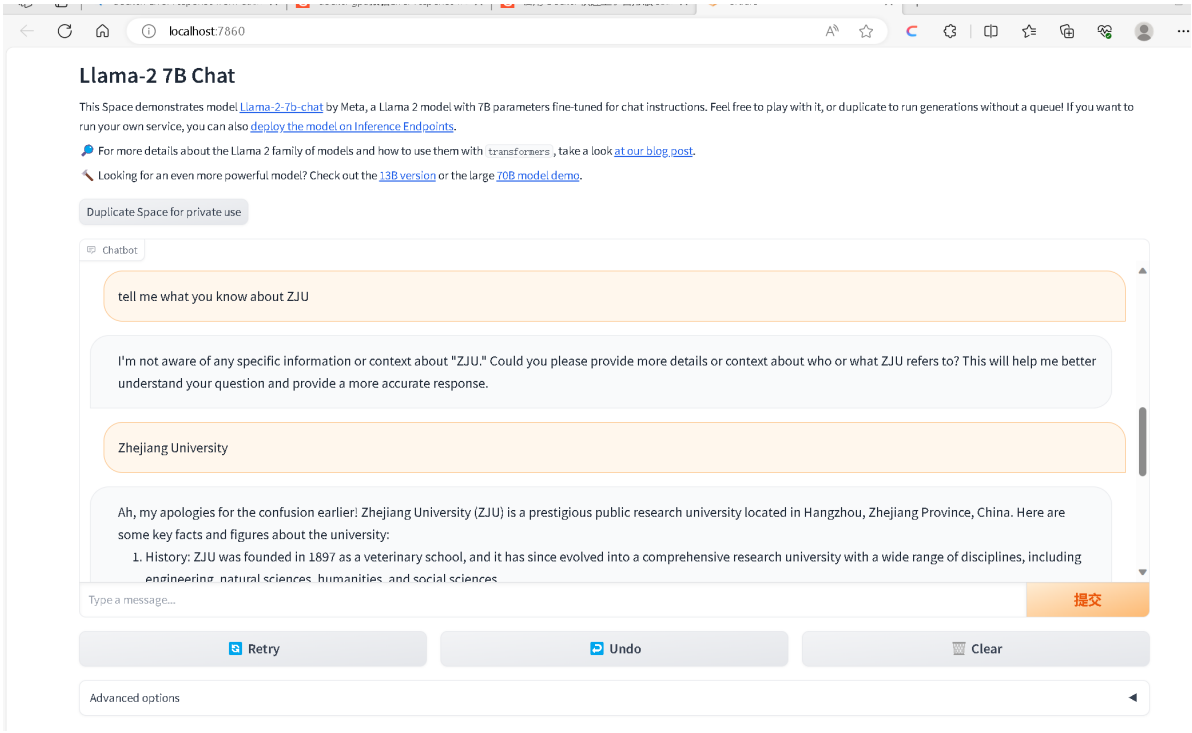
然后使用浏览器打开https://localhost:7860即可运行。

如果是使用vscode远程开发，会先出现下图的提示：



如果端口7860不可用，则可以修改为7865或者其他端口，如上图就修改为外部7865端口映射docker运行的7860端口。

然后vscode会提示是否需要转发端口上的内容，可以添加7865端口进入转发设置，然后打开任意浏览器使用llama2。



4. 参考链接

1. 最详细的ubuntu 安装 docker教程 - 知乎 (zhihu.com)
2. 使用 Docker 快速上手官方版 LLaMA2 开源大模型_soulteary的博客-CSDN博客
3. 【nvidia-smi报错】Failed to initialize NVML: Driver/library version mismatch-CSDN博客
4. [docker gpu报错Error response from daemon: could not select device driver “” with capabilities: [[gpu]]_Adenialzz的博客-CSDN博客](https://blog.csdn.net/weixin_44966641/article/details/123760614)
5. hf-mirror.com - Huggingface 镜像站

written by Hongye Fu 2023.11.24，感谢*soulteary*前辈的文档和仓库