讲义地址

https://docs.qq.com/doc/p/0555f97a90b6f2f9f039bd0c9251f5d7d1e7a506
密码:hsw123456

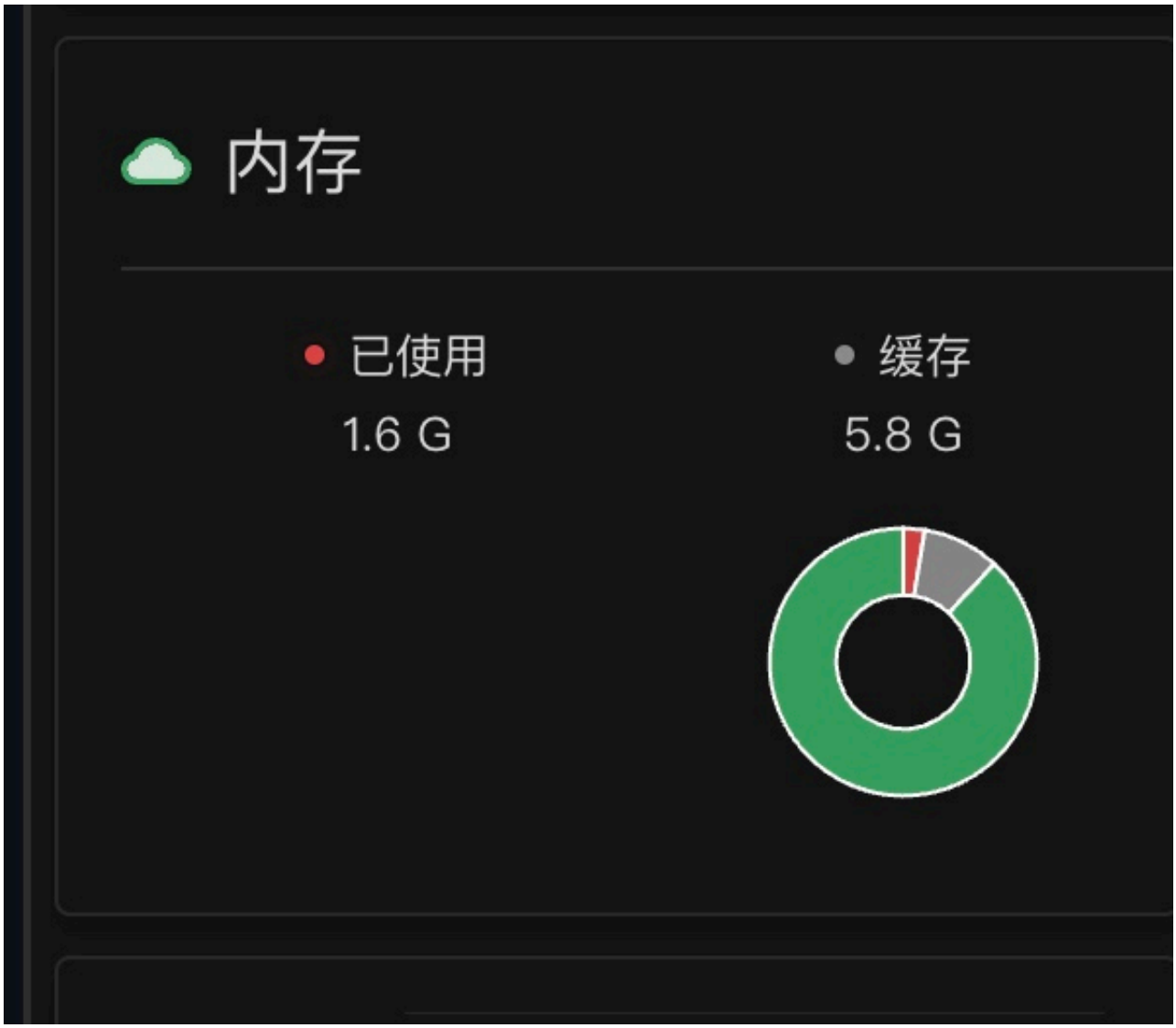
阿里云租赁GPU服务器

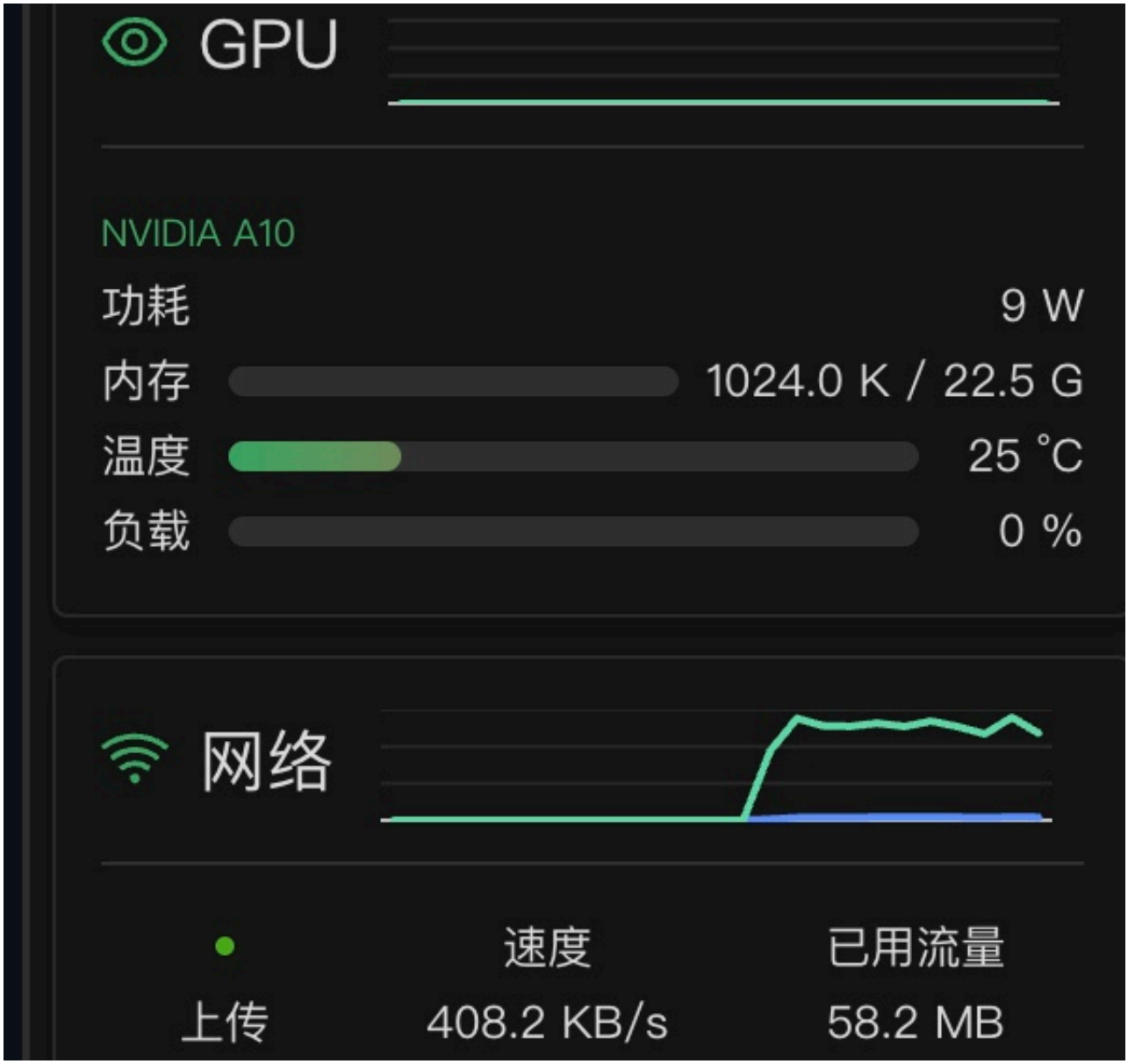
阿里云租赁GPU服务器：`https://ecs-buy.aliyun.com/ecs?spm=5176.8300896.J_5253785160.3.251e3680LsIXpx#/custom/postpay/cn-hangzhou?toNewVersion=true&instanceType=ecs.gn7i-c16g1.4xlarge`

之所以不用autoDL是因为它的每一个服务器实例其实都是一个docker容器，而docker实例无法套娃，做不了本次实验。

租赁阿里云服务器的时候，一定要注意看配置，必须是带gpu显卡的，并且系统盘内存最好是大一些，我选的是一个T,因为后续我们下载大模型文件，可能会耗费大量内存。

我是用xterminer来利用ssh连的服务器，必须要在监控模块看到gpu才算对。如下图：





安装docker和docker-compose

安装过程就省略了，最后一定要验证 `docker --version` 和 `docker-compose --version`，能看到版本号才能说明装完了。

准备compose配置文件

```
mkdir fastgpt
cd fastgpt
```

内容为：

```
# 数据库的默认账号和密码仅首次运行时设置有效
# 如果修改了账号密码，记得改数据库和项目连接参数，别只改一处~
# 该配置文件只是给快速启动，测试使用。正式使用，记得务必修改账号密码，以及调整合适的知识库参
```

```
数，共享内存等。
# 如何无法访问 dockerhub 和 git，可以用阿里云（阿里云没有arm包）

version: '3.3'
services:
  # db
  pg:
    #image: pgvector/pgvector:0.7.0-pg15 # docker hub
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/pgvector:v0.7.0 # 阿里云
    container_name: pg
    restart: always
    ports: # 生产环境建议不要暴露
      - 5432:5432
    networks:
      - fastgpt
    environment:
      # 这里的配置只有首次运行生效。修改后，重启镜像是不会生效的。需要把持久化数据删除再重启，
才有效果
      - POSTGRES_USER=username
      - POSTGRES_PASSWORD=password
      - POSTGRES_DB=postgres
    volumes:
      - ./pg/data:/var/lib/postgresql/data
  mongo:
    #image: mongo:5.0.18 # dockerhub
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/mongo:5.0.18 # 阿里云
    # image: mongo:4.4.29 # cpu不支持AVX时候使用
    container_name: mongo
    restart: always
    ports:
      - 27017:27017
    networks:
      - fastgpt
    command: mongod --keyFile /data/mongodb.key --replSet rs0
    environment:
      - MONGO_INITDB_ROOT_USERNAME=myusername
      - MONGO_INITDB_ROOT_PASSWORD=mypassword
    volumes:
      - ./mongo/data:/data/db
    entrypoint:
      - bash
      - -c
      - |
        openssl rand -base64 128 > /data/mongodb.key
        chmod 400 /data/mongodb.key
        chown 999:999 /data/mongodb.key
        echo 'const isInited = rs.status().ok === 1
        if(!isInited){
          rs.initiate({
              _id: "rs0",
              members: [
                  { _id: 0, host: "mongo:27017" }
```

```
              ]
          })
        }' > /data/initReplicaSet.js
        # 启动MongoDB服务
        exec docker-entrypoint.sh "$$@" &

        # 等待MongoDB服务启动
        until mongo -u myusername -p mypassword --authenticationDatabase admin --eval "print('waited for connection')" > /dev/null 2>&1; do
          echo "Waiting for MongoDB to start..."
          sleep 2
        done

        # 执行初始化副本集的脚本
        mongo -u myusername -p mypassword --authenticationDatabase admin /data/initReplicaSet.js

        # 等待docker-entrypoint.sh脚本执行的MongoDB服务进程
        wait $$!

  # fastgpt
  sandbox:
    container_name: sandbox
    # image: ghcr.io/labring/fastgpt-sandbox:latest # git
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/fastgpt-sandbox:latest # 阿里云
    networks:
      - fastgpt
    restart: always
  fastgpt:
    container_name: fastgpt
    # image: ghcr.io/labring/fastgpt:v4.8.8-fix2 # git
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/fastgpt:v4.8.8-fix2 # 阿里云
    ports:
      - 3000:3000
    networks:
      - fastgpt
    depends_on:
      - mongo
      - pg
      - sandbox
    restart: always
    environment:
      # root 密码，用户名为: root。如果需要修改 root 密码，直接修改这个环境变量，并重启即可。
      - DEFAULT_ROOT_PSW=1234
      # AI模型的API地址哦。务必加 /v1。这里默认填写了OneApi的访问地址。
      - OPENAI_BASE_URL=http://oneapi:3000/v1
      # AI模型的API Key。（这里默认填写了OneAPI的快速默认key，测试通后，务必及时修改）
      - CHAT_API_KEY=sk-fastgpt
      # 数据库最大连接数
      - DB_MAX_LINK=30
```

```
      # 登录凭证密钥
      - TOKEN_KEY=any
      # root的密钥，常用于升级时候的初始化请求
      - ROOT_KEY=root_key
      # 文件阅读加密
      - FILE_TOKEN_KEY=filetoken
      # MongoDB 连接参数. 用户名myusername,密码mypassword。
      - MONGODB_URI=mongodb://myusername:mypassword@mongo:27017/fastgpt?
authSource=admin
      # pg 连接参数
      - PG_URL=postgresql://username:password@pg:5432/postgres
      # sandbox 地址
      - SANDBOX_URL=http://sandbox:3000
      # 日志等级: debug, info, warn, error
      - LOG_LEVEL=info
      - STORE_LOG_LEVEL=warn
    volumes:
      - ./config.json:/app/data/config.json

  # oneapi
  mysql:
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/mysql:8.0.36 # 阿里云
    # image: mysql:8.0.36
    container_name: mysql
    restart: always
    ports:
      - 3306:3306
    networks:
      - fastgpt
    command: --default-authentication-plugin=mysql_native_password
    environment:
      # 默认root密码，仅首次运行有效
      MYSQL_ROOT_PASSWORD: oneapimmysql
      MYSQL_DATABASE: oneapi
    volumes:
      - ./mysql:/var/lib/mysql
  oneapi:
    container_name: oneapi
    # image: ghcr.io/songquanpeng/one-api:v0.6.7
    image: registry.cn-hangzhou.aliyuncs.com/fastgpt/one-api:v0.6.6 # 阿里云
    ports:
      - 3001:3000
    depends_on:
      - mysql
    networks:
      - fastgpt
    restart: always
    environment:
      # mysql 连接参数
      - SQL_DSN=root:oneapimmysql@tcp(mysql:3306)/oneapi
      # 登录凭证加密密钥
      - SESSION_SECRET=oneapikey
```

```
      # 内存缓存
      - MEMORY_CACHE_ENABLED=true
      # 启动聚合更新，减少数据交互频率
      - BATCH_UPDATE_ENABLED=true
      # 聚合更新时长
      - BATCH_UPDATE_INTERVAL=10
      # 初始化的 root 密钥（建议部署完后更改，否则容易泄露）
      - INITIAL_ROOT_TOKEN=fastgpt
    volumes:
      - ./oneapi:/data
networks:
  fastgpt:
```

准备fastgpt的配置文件

同样在fastgpt目录下，创建文件 config.json

内容如下：

```
{
  "feConfigs": {
    "lafEnv": "https://laf.dev"
  },
  "systemEnv": {
    "vectorMaxProcess": 15,
    "qaMaxProcess": 15,
    "pgHNSWEfSearch": 100
  },
  "llmModels": [
    {
        "model": "llama-3.1-instruct",
        "name": "llama-3.1-instruct",
        "maxContext": 128000,
        "maxResponse": 128000,
        "quoteMaxToken": 32000,
        "maxTemperature": 1.2,
      "charsPointsPrice": 0,
      "censor": false,
      "vision": false,
      "datasetProcess": false,
      "usedInClassify": true,
      "usedInExtractFields": true,
      "usedInToolCall": true,
      "usedInQueryExtension": true,
      "toolChoice": false,
      "functionCall": false,
      "customCQPrompt": "",
      "customExtractPrompt": "",
      "defaultSystemChatPrompt": "",
```

```
    "defaultConfig": {}
  },
  {
    "model": "qwen2-instruct",
    "name": "qwen2-instruct",
    "avatar": "/imgs/model/qwen.svg",
    "maxContext": 128000,
    "maxResponse": 128000,
    "quoteMaxToken": 32000,
    "maxTemperature": 1.2,
    "charsPointsPrice": 0,
    "censor": false,
    "vision": false,
    "datasetProcess": true,
    "usedInClassify": true,
    "usedInExtractFields": true,
    "usedInToolCall": true,
    "usedInQueryExtension": true,
    "toolChoice": false,
    "functionCall": false,
    "customCQPrompt": "",
    "customExtractPrompt": "",
    "defaultSystemChatPrompt": "",
    "defaultConfig": {}
  },
  {
    "model": "glm-4v",
    "name": "glm-4v",
    "avatar": "/imgs/model/chatglm.svg",
    "maxContext": 128000,
    "maxResponse": 128000,
    "quoteMaxToken": 32000,
    "maxTemperature": 1.2,
    "charsPointsPrice": 0,
    "censor": false,
    "vision": true,
    "datasetProcess": true,
    "usedInClassify": true,
    "usedInExtractFields": true,
    "usedInToolCall": true,
    "usedInQueryExtension": true,
    "toolChoice": true,
    "functionCall": false,
    "customCQPrompt": "",
    "customExtractPrompt": "",
    "defaultSystemChatPrompt": "",
    "defaultConfig": {}
  },
  {
    "model": "ERNIE-Speed-128K",
    "name": "ERNIE-Speed-128K",
  "avatar": "/imgs/model/ernie.svg",
```

```
      "maxContext": 128000,
      "maxResponse": 128000,
      "quoteMaxToken": 32000,
      "maxTemperature": 1.2,
      "charsPointsPrice": 0,
      "censor": false,
      "vision": false,
      "datasetProcess": true,
      "usedInClassify": true,
      "usedInExtractFields": true,
      "usedInToolCall": true,
      "usedInQueryExtension": true,
      "toolChoice": true,
      "functionCall": false,
      "customCQPrompt": "",
      "customExtractPrompt": "",
      "defaultSystemChatPrompt": "",
      "defaultConfig": {}
    },
    {
      "model": "gpt-4o-mini",
      "name": "gpt-4o-mini",
      "avatar": "/imgs/model/openai.svg",
      "maxContext": 125000,
      "maxResponse": 4000,
      "quoteMaxToken": 120000,
      "maxTemperature": 1.2,
      "charsPointsPrice": 0,
      "censor": false,
      "vision": true,
      "datasetProcess": true,
      "usedInClassify": true,
      "usedInExtractFields": true,
      "usedInToolCall": true,
      "usedInQueryExtension": true,
      "toolChoice": true,
      "functionCall": false,
      "customCQPrompt": "",
      "customExtractPrompt": "",
      "defaultSystemChatPrompt": "",
      "defaultConfig": {}
    },
    {
      "model": "gpt-4o",
      "name": "gpt-4o",
      "avatar": "/imgs/model/openai.svg",
      "maxContext": 125000,
      "maxResponse": 4000,
      "quoteMaxToken": 120000,
      "maxTemperature": 1.2,
      "charsPointsPrice": 0,
      "censor": false,
```

```
      "vision": true,
      "datasetProcess": false,
      "usedInClassify": true,
      "usedInExtractFields": true,
      "usedInToolCall": true,
      "usedInQueryExtension": true,
      "toolChoice": true,
      "functionCall": false,
      "customCQPrompt": "",
      "customExtractPrompt": "",
      "defaultSystemChatPrompt": "",
      "defaultConfig": {}
    }
  ],
  "vectorModels": [
    {
      "model": "m3e-base",
      "name": "m3e-base",
      "charsPointsPrice": 0,
      "defaultToken": 256,
      "maxToken": 512,
      "weight": 100,
      "defaultConfig": {},
      "dbConfig": {},
      "queryConfig": {}
    },
    {
      "model": "text-embedding-ada-002",
      "name": "Embedding-2",
      "avatar": "/imgs/model/openai.svg",
      "charsPointsPrice": 0,
      "defaultToken": 700,
      "maxToken": 3000,
      "weight": 100,
      "defaultConfig": {},
      "dbConfig": {},
      "queryConfig": {}
    },
    {
      "model": "text-embedding-3-large",
      "name": "text-embedding-3-large",
      "avatar": "/imgs/model/openai.svg",
      "charsPointsPrice": 0,
      "defaultToken": 512,
      "maxToken": 3000,
      "weight": 100,
      "defaultConfig": {
        "dimensions": 1024
      }
    },
    {
      "model": "embeding3",
```

```
      "name": "embeding3",
      "avatar": "/imgs/model/chatglm.svg",
      "charsPointsPrice": 0,
      "defaultToken": 512,
      "maxToken": 3000,
      "weight": 100
    }
  ],
  "reRankModels": [
    {
      "model": "bge-reranker-v2-m3",
      "name": "bge-reranker-v2-m3"
    }
  ],
  "audioSpeechModels": [
    {
      "model": "tts-1",
      "name": "OpenAI TTS1",
      "charsPointsPrice": 0,
      "voices": [
        { "label": "Alloy", "value": "alloy", "bufferId": "openai-Alloy" },
        { "label": "Echo", "value": "echo", "bufferId": "openai-Echo" },
        { "label": "Fable", "value": "fable", "bufferId": "openai-Fable" },
        { "label": "Onyx", "value": "onyx", "bufferId": "openai-Onyx" },
        { "label": "Nova", "value": "nova", "bufferId": "openai-Nova" },
        { "label": "Shimmer", "value": "shimmer", "bufferId": "openai-Shimmer" }
      ]
    }
  ],
  "whisperModel": {
    "model": "whisper-1",
    "name": "Whisper1",
    "charsPointsPrice": 0
  }
}
```

如果改变了这个文件的内容，想要让他立即生效的话，只需要 `docker restart fastgpt`

启动系统

进入fastgpt目录，并且执行 `docker-compose up -d`

出现下面的结果说明在正常安装。这里会下载很多组件，会花一些时间。

```
终端 3 ×  +
Last login: Thu Oct 17 20:01:02 2024 from 120.228.142.167
root@iZbp13vmycyz2y29guorq1Z:~# cd fastgpt/
root@iZbp13vmycyz2y29guorq1Z:~/fastgpt# docker-compose up -d
Creating network "fastgpt_fastgpt" with the default driver
Pulling pg (registry.cn-hangzhou.aliyuncs.com/fastgpt/pgvector:v0.7.0)...

v0.7.0: Pulling from fastgpt/pgvector
b0a0cf830b12: Pulling fs layer
b311dac095c3: Downloading [==================================================>]  1.165kB/1.1
b311dac095c3: Download complete
b0a0cf830b12: Downloading [>                                                  ]  296.9kB/29.
15MB3e66f657: Pulling fs layer
fe401a45083b: Downloading [>                                                  ]   19.8kB/1.4
b0a0cf830b12: Downloading [=>                                                 ]  881.4kB/29.
15MBd9a95b21: Waiting
fe401a45083b: Downloading [==================>                               ]  562.8kB/1.4
36163cea98c3: Downloading [============>                                     ]  1.151MB/4.5
b0a0cf830b12: Downloading [==>                                                ]   1.47MB/29.
15MB1a45083b: Downloading [=======================================>          ]  1.139MB/1.4
b0a0cf830b12: Downloading [===>                                               ]  2.059MB/29.
15MB3cea98c3: Downloading [==================>                               ]  1.759MB/4.5
b0a0cf830b12: Downloading [====>                                              ]  2.649MB/29.
15MB3cea98c3: Downloading [=========================>                        ]   2.32MB/4.5
33MB
b0a0cf830b12: Downloading [=====>                                             ]  3.234MB/29.
15MB

36163cea98c3: Downloading [=====================================>            ]  3.504MB/4.5
b0a0cf830b12: Downloading [======>                                            ]  3.825MB/29.
15MB3cea98c3: Downloading [=======================================>          ]  4.099MB/4.5
b0a0cf830b12: Downloading [=======>                                           ]  4.412MB/29.
15MB
b0a0cf830b12: Pull complete
b311dac095c3: Pull complete
36163cea98c3: Pull complete
fe401a45083b: Pull complete
b4f63e66f657: Pull complete
9a8245430c33: Pull complete
74c893d239e5: Pull complete
789ed9a95b21: Pull complete
5b691e4e12b4: Downloading [================================================>  ]  103.6MB/106
.9MBd5360f88: Download complete
0292671f3c2d: Download complete
5247a27db111: Download complete
b3a42a5269c3: Download complete
5fc19b802589: Download complete
973f95d4f335: Download complete
694339e21380: Download complete
.5kB
```

出现下面这个说明启动完成：

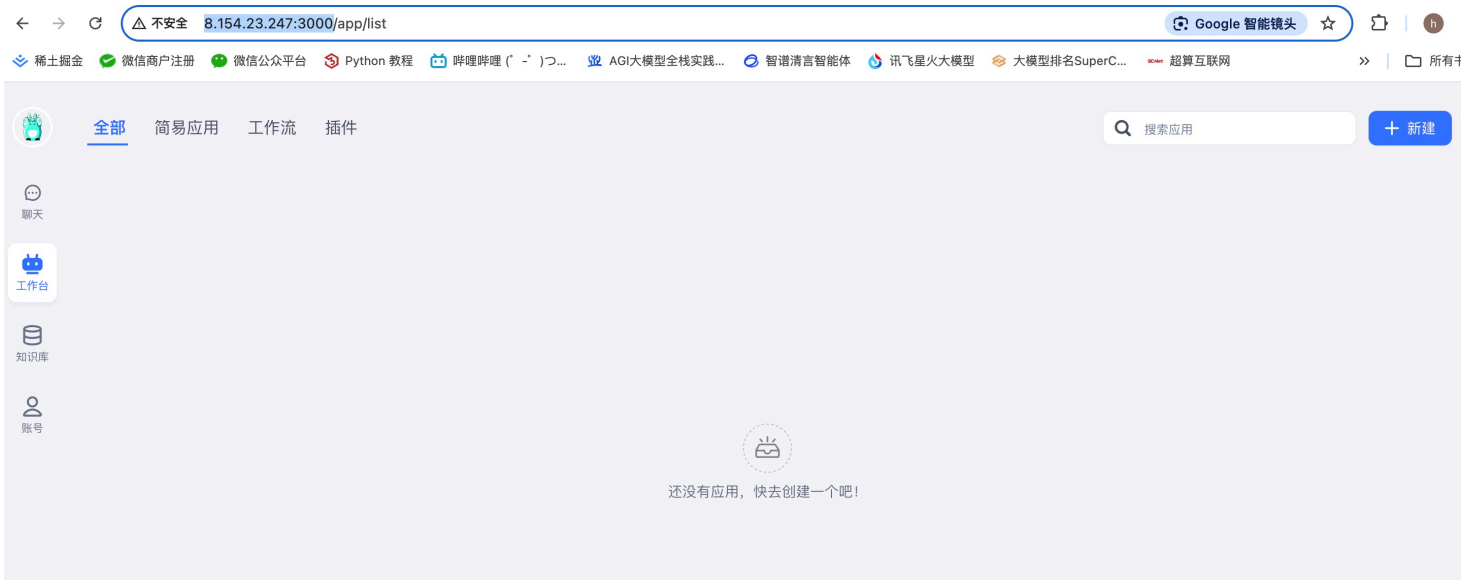
```
Digest: sha256:480a8d63af5057f7d81437dc138da941998657779b6c0150fd5e595b6ff72c71
Status: Downloaded newer image for registry.cn-hangzhou.aliyuncs.com/fastgpt/one-api:v0.6.6
Creating sandbox ... done
Creating mysql   ... done
Creating mongo   ... done
Creating pg      ... done
Creating oneapi  ... done
Creating fastgpt ... done
```

用 docker-compose ps

```
root@iZbp13vmycyz2y29guorq1Z:~/fastgpt# docker-compose ps
 Name                 Command                State                        Ports
-------------------------------------------------------------------------------------------------
fastgpt   sh -c node --max-old-space ...   Up      0.0.0.0:3000->3000/tcp,:::3000->3000/tcp
mongo     bash -c openssl rand -base ...   Up      0.0.0.0:27017->27017/tcp,:::27017->27017/tcp
mysql     docker-entrypoint.sh --def ...   Up      0.0.0.0:3306->3306/tcp,:::3306->3306/tcp,
                                                   33060/tcp
oneapi    /one-api                         Up      0.0.0.0:3001->3000/tcp,:::3001->3000/tcp
pg        docker-entrypoint.sh postgres    Up      0.0.0.0:5432->5432/tcp,:::5432->5432/tcp
sandbox   docker-entrypoint.sh node  ...   Up
```

本机浏览器中输入：http://8.154.23.247:3000/

如果能得到如下画面：



输入：http://8.154.23.247:3001/ 如果能得到这个：

就说明系统启动正常。

安装 xinference

xinference是一个傻瓜式操作的大模型部署平台，有ui交互界面。

回到root，执行如下命令:

```
mkdir xinference
cd xinference
docker run \
  -d \
  -e XINFERENCE_MODEL_SRC=modelscope \
  -v $(pwd)/.xinference:/root/.xinference \
  -v $(pwd)/.cache/huggingface:/root/.cache/huggingface \
  -v $(pwd)/.cache/modelscope:/root/.cache/modelscope \
  -p 9997:9997 \
  --gpus all \
  registry.cn-hangzhou.aliyuncs.com/xprobe_xinference/xinference \
  xinference-local -H 0.0.0.0
```

这里它会下载很多大文件，所以要求服务器内存一定要够大。

如果命令执行出现问题，可以尝试安装 nvidia container toolkit:

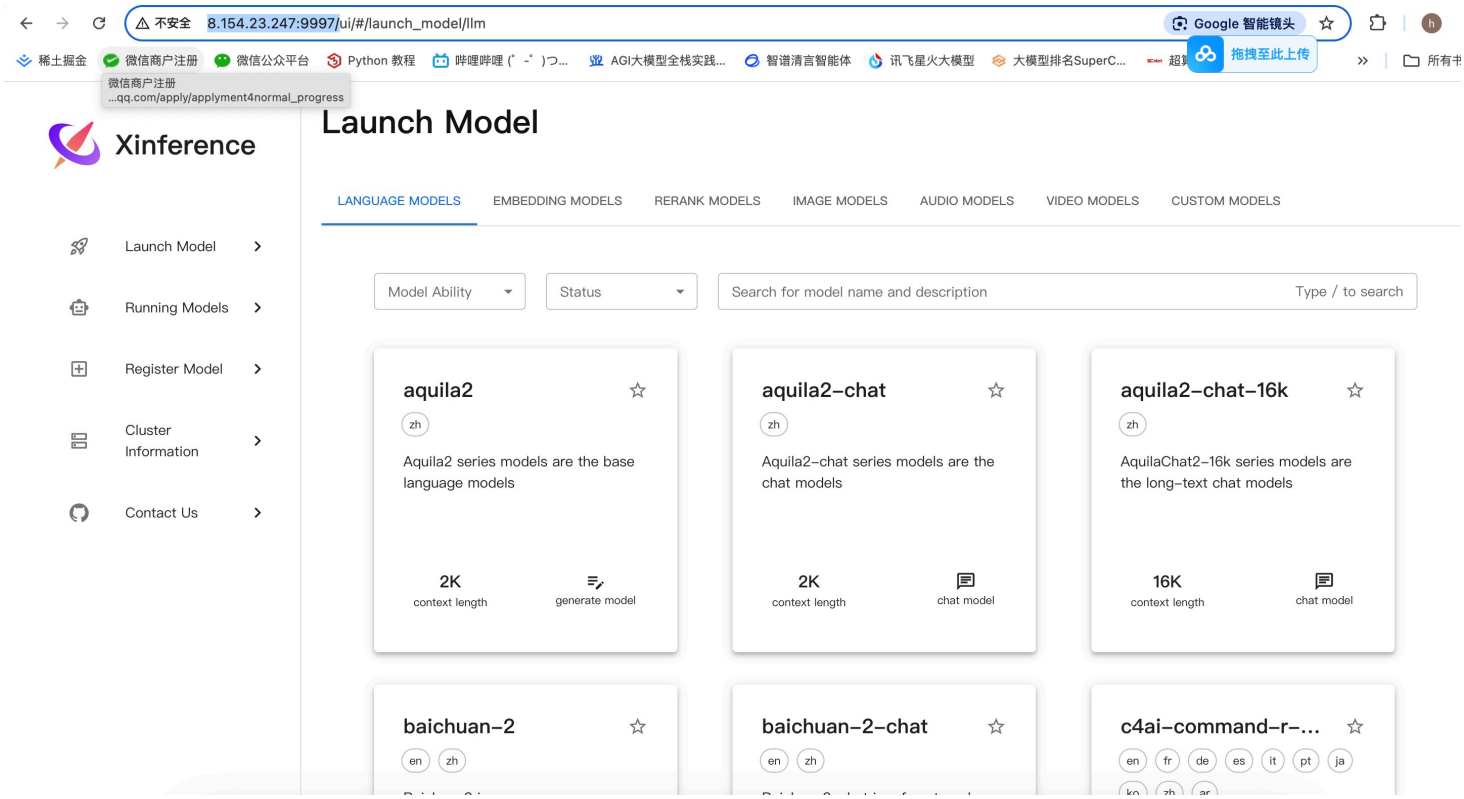
```
sudo apt install curl -y
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg --
dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg && curl -s -
L https://nvidia.github.io/libnvidia-container/stable/deb/nvidia-container-
toolkit.list | sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-
container-toolkit-keyring.gpg] https://#g' | sudo tee
/etc/apt/sources.list.d/nvidia-container-toolkit.list
sudo apt-get update
sudo apt-get install -y nvidia-container-toolkit
```

这个命令执行完了之后要重启docker: `sudo systemctl restart docker`，之后再次执行第一个命令。

用 `docker ps` 能看到如下图所示，说明启动成功:

```
root@iZbp13vmycyz2y29guorq1Z:~/fastgpt/xinference/xinference/xinference/xinference# docker ps
CONTAINER ID   IMAGE                                                                  COMMAND                  CREATED              STATUS
          PORTS                                                   NAMES
76c183e30e56   registry.cn-hangzhou.aliyuncs.com/xprobe_xinference/xinference     "xinference-local -H…"   About a minute ago   Up About a
 minute   0.0.0.0:9997->9997/tcp, :::9997->9997/tcp              inspiring_hermann
```

在我本机浏览器中输入: `http://8.154.23.247:9997/` 如果出现:



则说明启动xinference成功。

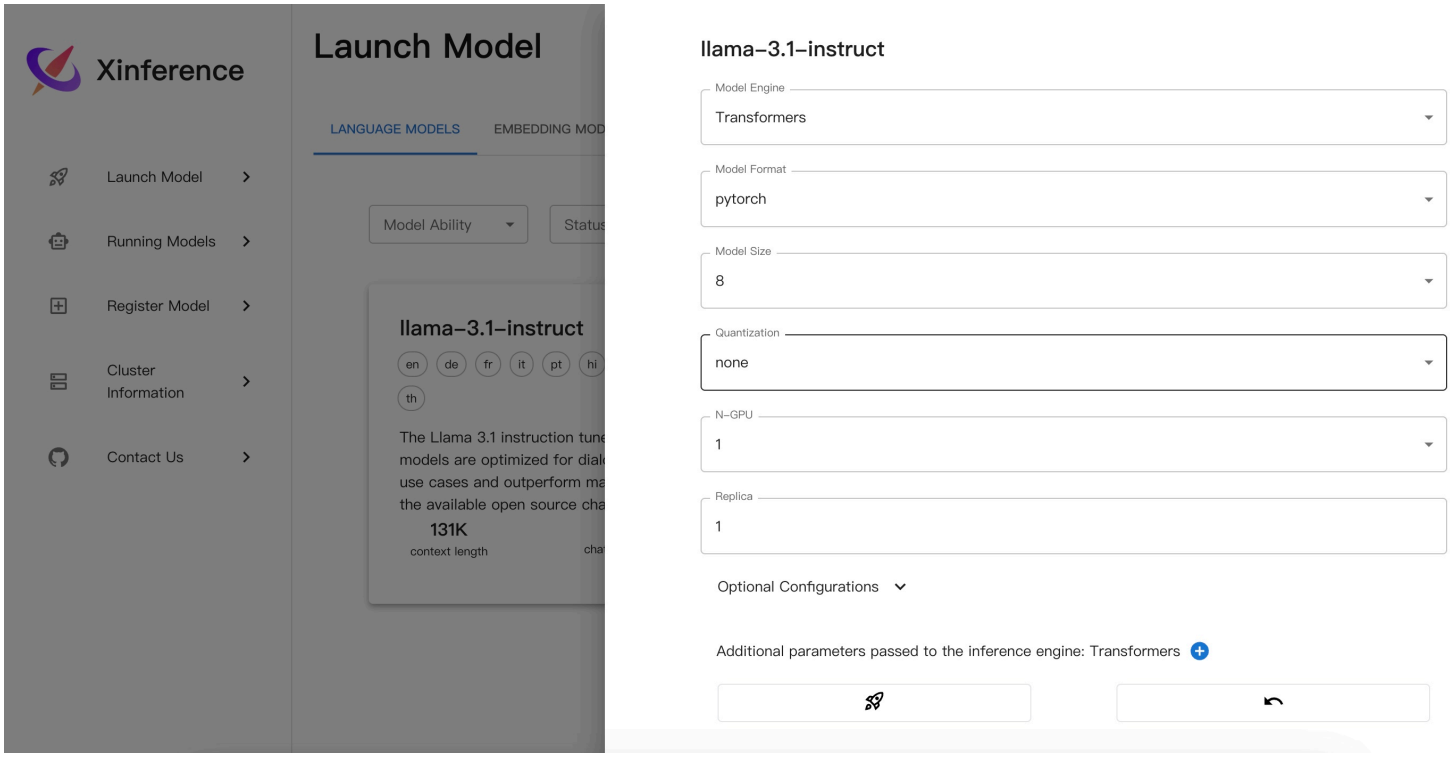
至此作业所需的所有准备工作都完毕。

部署本地模型

首先在xinference交互界面中，我们部署3个模型，这3个都是接下来的实验必须的。

- chat模型
  llama-3.1-instruct
- embeding模型
  m3e-base
- rerank模型
  bge-reranker-v2-m3

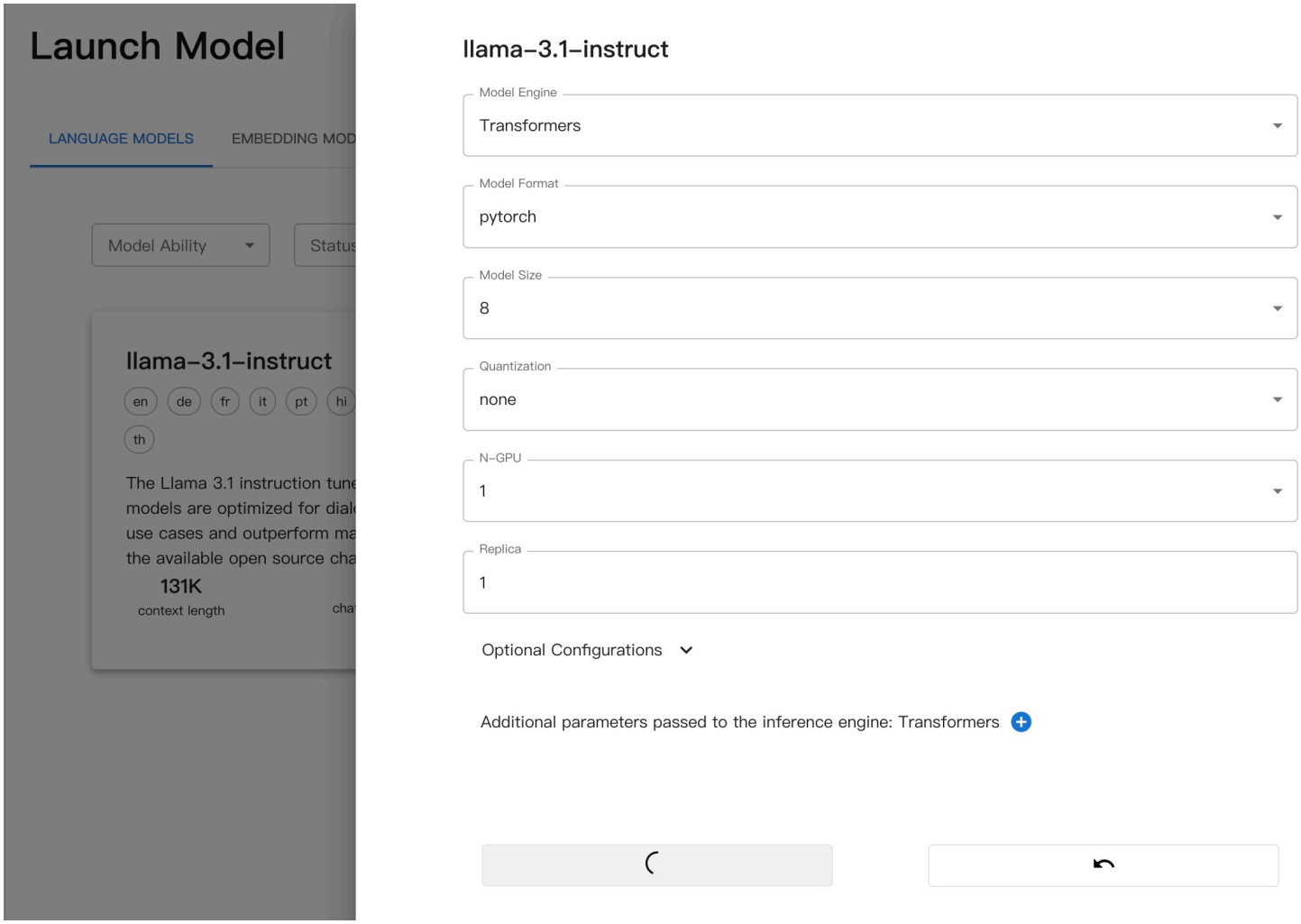
以第一个为例，在模型这一栏搜索 llama-3.1-instruct，点进去，然后按照下面图的逐个配置：



点击下面的小火箭图标，系统就会自动给你下载并且部署。

但是很不幸，我的xinference无法下载所有模型。重启xinference之后问题解决，从服务器的带宽监控来看，正在下载，耐心等待。

刚才的小火箭图标也变成了loading：



等他下完吧。这会可千万别手贱，不然会前功尽弃。刚才就是因为动了一下，才频繁报错的。上面是在modelscope（大模型国内镜像源）去下载，所以应该不存在网络问题。道理我都懂，为什么这里不显示下载进度？？？

这3个模型已经都在running状态了。

LANGUAGE MODELS | EMBEDDING MODELS | RERANK MODELS | IMAGE MODELS | AUDIO MODELS | VIDEO MODELS | FLEXIBLE MODELS

ID	Name	Address	GPU Indexes	Size	Quantization	Replica	Actions
llama-3.1-instruct	llama-3.1-inst...	0.0.0.0:36441	0	8	none	1	

LANGUAGE MODELS | EMBEDDING MODELS | RERANK MODELS | IMAGE MODELS | AUDIO MODELS | VIDEO MODELS | FLEXIBLE MODELS

ID	Name	Address	GPU Indexes	Replica	Actions
m3e-base	m3e-base	0.0.0.0:35039	0	1	

LANGUAGE MODELS | EMBEDDING MODELS | RERANK MODELS | IMAGE MODELS | AUDIO MODELS | VIDEO MODELS | FLEXIBLE MODELS

ID	Name	Address	GPU Indexes	Replica	Actions
bge-reranker-v2-m3	bge-reranker-v2-m3	0.0.0.0:35995	0	1	

将本地模型加入到oneapi渠道中

注意这里的 8.154.23.247是我的服务器公网ip地址。下面模型栏，填的是刚部署的3个大模型。

经过我的测试，发现llama-3.1-instruct这个模型回答存在问题，总是给我一个json，决定重新下一个 `baichuan-2-chat` 部署上去。

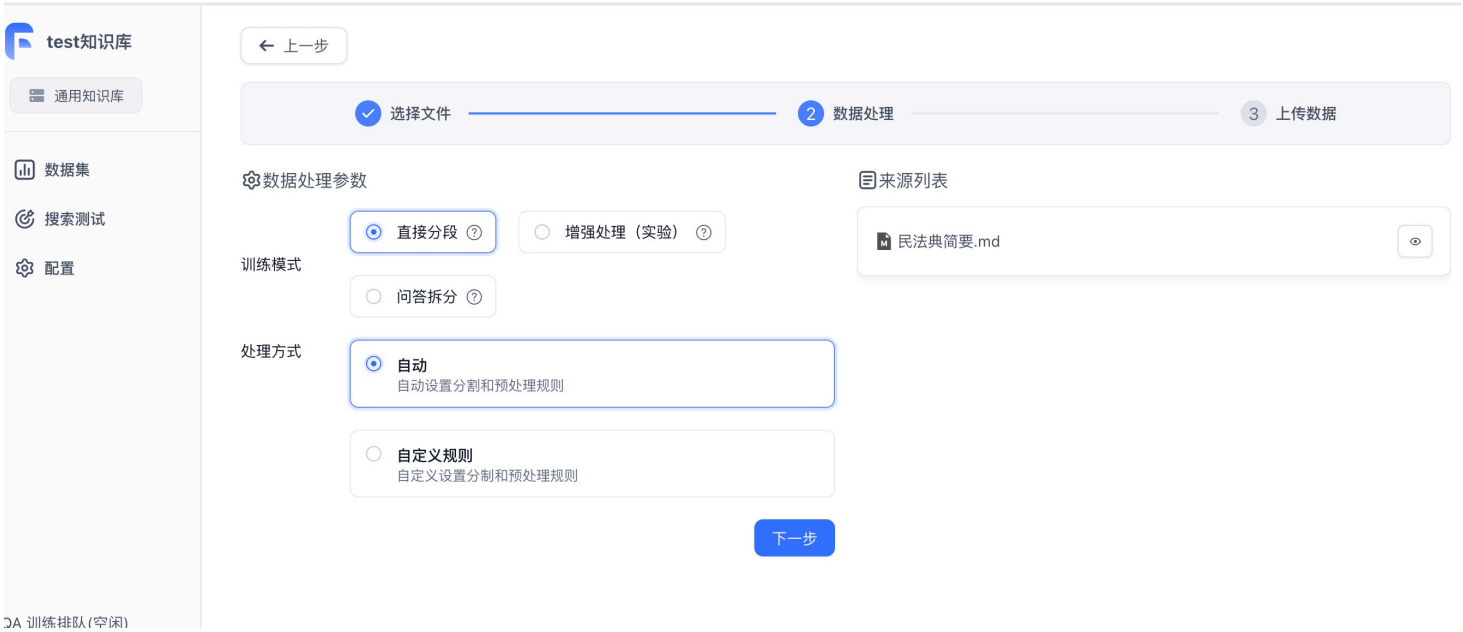
fastgpt创建对话

创建对话，并选择刚才的baichuan-2-chat，看下图，已经可以正常聊天。

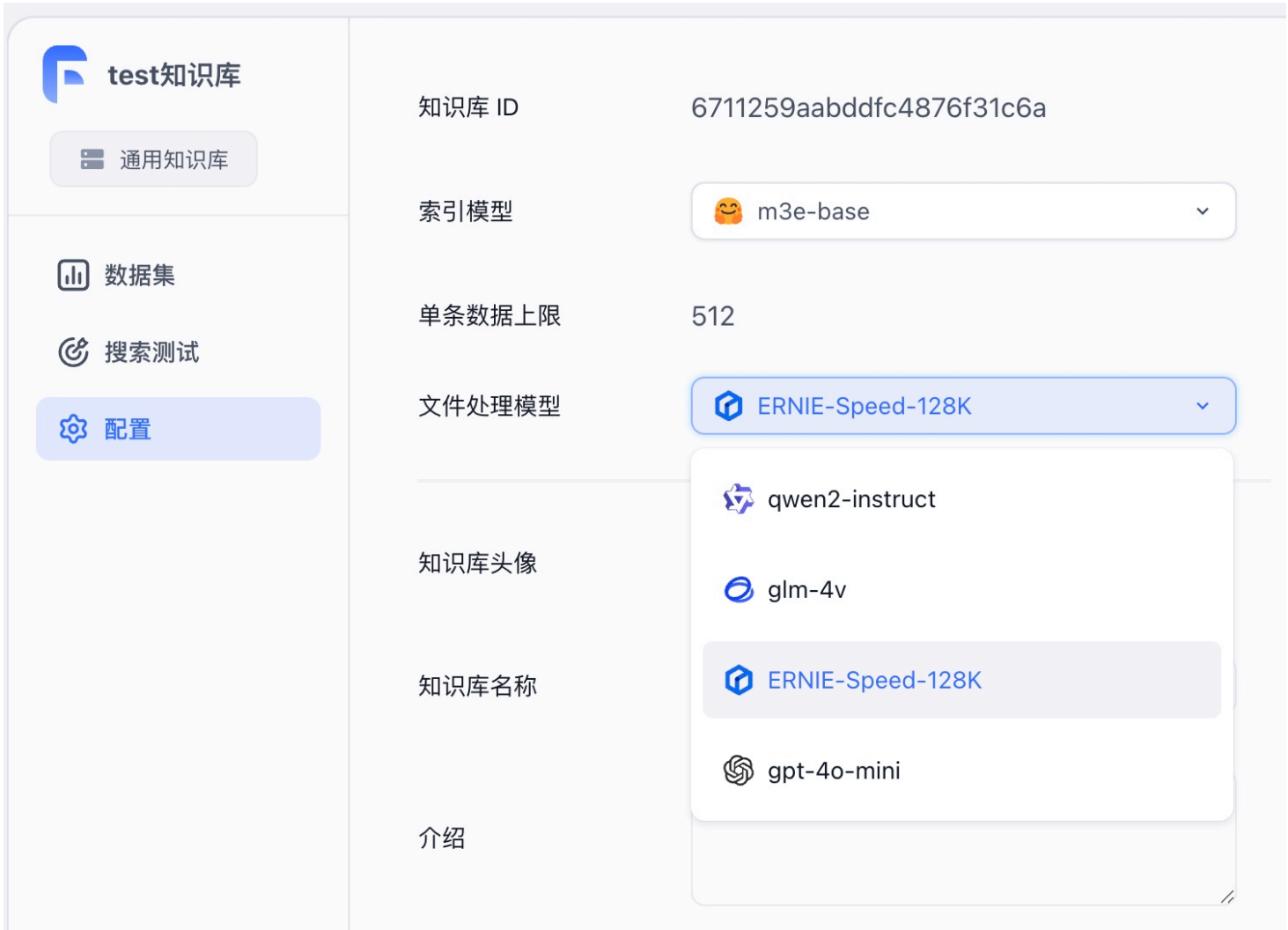
附加知识库

知识库就绪的情况如下

在附加知识库的过程中，有一些配置会影响知识库是否正常。
比如下面的，选择直接分段，可能导致知识库无法正常就绪，尝试更换为 直接分段（猜测这个和文档结构有关，总之要想办法变更配置让 知识库可以正常使用。）



另一个需要注意的是，下图中有两个配置会影响到知识库能否正常加载：
第一个就是 m3e-base 索引模型，这个模型在 **安装xinference**这一章节已经部署了，所以我们就可以选它。
另一个就是文件处理模型，我们这里选择的是 ERNIE-Speed-128K（这个是我按照讲义的步骤，在 百度智能云平台上注册的一个应用，然后在oneapi中添加了此渠道，所以我可以用它来帮我处理文件内容）



最后的测试

知识库就绪，本地模型就绪。现在来进行测试。

在我们新建的机器人配置中，选择 我们本地部署的 baichuan-2-chat，同时关联刚才加进去的知识库，可以看到下图右下角的回答中，已经在饮用我上传的pdf文件内容了。

实验成功，下一步把聊天机器人发布出去。

甚至可以发布成一个网页给大众使用。

http://8.154.23.247:3000/chat/share?shareId=19nfzixgztcy2hdg87yz6yqx

只不过正式发布出去之后，在回答中不会像测试过程中看到原文的引用。

实验成功。

记得把按量计费的阿里云服务器关机，挺贵的。