1. Первым делом нужно зарегистрироваться на Hugging Face и получить индивидуальный токен для чтения файлов. Если этого не сделать, могут возникнуть проблемы с установкой файлов некоторых моделей.

2. Далее установка необходимых пакетов.
   Я устанавливала из PyPi, используя WSL терминал.

   - Установка Rust:

```
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
source $HOME/.cargo/env
```

   Проверка версий:

```
rustc --version
cargo --version
```

   Если установка прошла успешно, в выводе после исполнения последних двух команд появится версия установленного Rust.

   - Следующим шагом нужно установить следующие библиотеки и утилиты:

```
sudo apt install libssl-dev
sudo apt install pkg-config
```

   Проверка установленных пакетов:

```
apt list --installed libssl-dev
apt list --installed pkg-config
```

   - В папке с проектом создаю окружение и активирую его:

```
python3 -m venv myenv
source myenv/bin/activate
```

   - Установка *mistrals* (в моем случае без ускорителей):

```
pip install mistralrs -v
```

3. Подготовка к работе с моделями.

   - Работу с проектом следует начать с авторизации на Hugging Face (здесь понадобится токен из п. 1):

```
pip install huggingface-hub
```

```
huggingface-cli login
```

После выполнения login команды нужно ввести свой токен.

- Клонирование репозитория:

```
git clone https://github.com/EricLBuehler/mistral.rs.git
```

- Перейти в папку проекта:

```
cd mistral.rs
```

- Запустить сборку проекта (у меня базовая команда, без использования ускорений):

```
cargo build --release
```

- Запуск проекта:

```
cargo run --release
```

- Копирование бинарного файла, созданного после сборки автоматически, в текущую директорию (я поменяла имя, т. к. из-за существующей директории с тем же именем, файл не копируется):

```
cp ./target/release/mistralrs-server ./mistralrs-server-executable
```

Проверка наличия:

```
ls -l ./mistralrs-server-executable
```

4. Теперь можно приступить к работе с LLM.
В этом пункте рассматривается установка модели и запуск в интерактивном режиме.

- Я загружала модель Phi 3 128k mini с квантованием через ISQ в Q4K.

```
./mistralrs-server-executable --token-source path:/home/daria/.cache/huggingface/token --isq Q4K -i plain -m microsoft/Phi-3-mini-128k-instruct -a phi3
```

Эта команда установит и запустит модель в интерактивном режиме, можно начинать общение с моделью. Для закрытия интерактивного режима использовать сочетание клавиш Ctrl + C.

- Команда для повторного запуска уже загруженной модели в интерактивном режиме:

```
./mistralrs-server-executable --isq Q4K -i plain -m microsoft/Phi-3-mini-128k-instruct -a phi3
```

5. Создание чат-бота.

- Установка необходимых библиотек:

```
pip install flask openai
```

- В главную папку проекта *mistral.rs* копирую файл *chatbot.py* из папки *examples*. Заменяю строку:
- `client = OpenAI(api_key="foobar", base_url="http://localhost:1234/v1/")`
  на:
- `client = OpenAI(api_key="foobar", base_url="http://localhost:8080/v1/")`

- Запуск предварительно установленной модели на локальном порту:

```
./mistralrs-server-executable --isq Q4K --port 8080 plain --model-id microsoft/Phi-3-mini-128k-instruct -a phi3
```

- Далее открыть второй терминал, активировать окружение, перейти в папку проекта *mistral.rs*. Следующая команда запустит чат-бота на предустановленной модели:

```
python chatbot.py
```

После запуска файла можно ввести промпт и начинать диалог с моделью:

```
Enter system prompt >>> you are a virtual assistant who helps with studies and provides emotional support

>>> Hi

Hello! I'm here to assist you with your studies or provide emotional support.

How can I help you today? Whether it's understanding complex concepts, preparing for exams, or just needing someone to talk to, feel free to share what's on your mind. Let's tackle those challenges together!

>>>
```

- Создание веб-интерфейса.
  На этом шаге я создаю бэкенд-файл *app.py* в главной папке проекта *mistral.rs*
- Создание HTML-шаблона.
  Далее создаю HTML файл с простым пользовательским интерфейсом для общения с чат-ботом.
  В основной папке проекта *mistral.rs* создаю папку templates, внутри *templates* создаю файл *index.html*.

- Далее открыть третий терминал, активировать окружение, перейти в папку проекта *mistral.rs*. После создания бэкэнд и фронтенд файлов можно запускать Flask-приложение.

```
python app.py
```

Важно: в первом терминале модель должна быть запущена на локальном порту, во втором – запущен файл *chatbot.py* (это было описано и сделано в пунктах выше).

После запуска *app.py* чат-бот будет запущен с пользовательским интерфейсом на локальном порту. Терминал выдаст сообщение, предлагающее перейти по ссылке:

```
Running on http://127.0.0.1:5000
```

6. Нужно перейти по локальной ссылке и можно начинать общение с ботом.