



Chief Boris

Персональный котошеф 🐾

Ссылка на бота:

https://t.me/Cook_for_youu_bot

Проект представляет собой персонального ассистента по приготовлению блюд в виде чат-бота, основанного на системе контролируемой генерации RAG и промпт-инжиниринга.

Цель проекта

Создать персонального ассистента на основе системы RAG и промпт-инжиниринга, который может подбирать рецепты по ингредиентам и вкусовым предпочтениям пользователя, а также учитывать его аллергии и вести гибкий диалог.



Задачи

1. Собрать базу рецептов
2. Нормализовать и перевести в нужный для retriever-a формат
3. Подобрать метод и реализовать retriever — систему поиска по БД рецептов
4. Настроить фильтрацию информации из выдачи retriever-a.
5. Подобрать оптимальную LLM для генератора.
6. Настроить генерацию ответа пользователю на основе полученных контекстов
7. Выбрать метод реализации чат-бота
8. Оформить удобный пользовательский интерфейс для взаимодействия

Данные

Датасет из 27 тысяч рецептов, содержащий в себе название, ингредиенты, процесс приготовления, фото и ссылку на рецепт.

```
Unnamed: 0                                     name \
0      0  рассольник классический с перловкой и солеными...
1      1  Суп пюре из белокочанной капусты
2      2  Постные щи из квашеной капусты
3      3  Тюря- простой суп быстро и вкусно
4      4  Фасолевый суп из красной фасоли

                                composition      cooking_type \
0  [{'Перловка': 0.1, 'unit': 'стак. (200 мл)'}, ...      варка, жарка
1  [{'Капуста белокочанная': 50.0, 'unit': 'гр'}, ...      варка
2  [{'Капуста квашеная': 116.7, 'unit': 'гр'}, {'...      варка, жарка, тушение
3  [{'Квас': 0.2, 'unit': 'л'}, {'Лук репчатый': ...      сырое
4  [{'Вода': 0.3, 'unit': 'л'}, {'Картошка': 0.3, ...      варка

                                Инструкции dish_type      Дата \
0  Подготовить указанные ингредиенты для приготов...      первое  05.06.2015
1  Необходимые ингредиенты\r\nНарезаем лук, морко...      первое  27.06.2015
2  Честно признаюсь, у меня не было репы на момен...      первое  12.02.2013
3  \r\nНачинаем мы приготовление тюри с того, что...      первое  02.03.2011
4  Подготовить ингредиенты. Для приготовления суп...      первое  28.01.2013

                                photo      source \
0  photo_1000menu_1.jpg  https://1000.menu/cooking/33395-rassolnik-s-pe...
1  photo_1000menu_2.jpg  https://1000.menu/cooking/25399-sup-pure-iz-be...
2  photo_1000menu_3.jpg  https://1000.menu/cooking/5159-postnje-shchi
3  photo_1000menu_4.jpg  https://1000.menu/cooking/5085-turya
4  photo_1000menu_5.jpg  https://1000.menu/cooking/38765-fasolevyi-sup-...

                                composition_inter
0  [{'product_id': 4253, 'name_source': 'Перловая...
1  [{'product_id': 2286, 'name_source': 'Капуста ...
2  [{'product_id': 0, 'name_source': 'Капуста ква...
3  [{'product_id': 0, 'name_source': 'Квас', 'uni...
4  [{'product_id': 828, 'name_source': 'Вода', 'u...
```

Предобработка

Оставляем только ключевые поля, содержащие важную информацию и убираем дубликаты.

```
[59] df = df.drop(columns=['Unnamed: 0', 'Дата', 'source', 'composition_inter', 'cooking_type'])
```

```
df.columns
```

```
Index(['name', 'composition', 'Инструкции', 'dish_type', 'photo'], dtype='object')
```

```
[61] df.columns = ['Название блюда', 'Ингредиенты', 'Приготовление', 'Тип блюда', 'Фото']
```

```
# Удаление дубликатов, сохраняя только первое вхождение  
df_cleaned = df.drop_duplicates()
```

Retriever

Модель эмбедингов: [intfloat/multilingual-e5-base](#)

- Оптимизирована под Retrieval (поиск): обучена на задаче поиска (retrieval) с парами query и passage. Это значительно повышает точность поиска.
- Поддерживает русский язык.
- Легко запускается на локальной машине.

Метод индексирования: [FAISS.IndexFlatL2](#)

- FAISS — стандарт в задачах семантического поиска, совместим с NumPy и легко расширяется.
- IndexFlatL2 — точный индекс (exhaustive search), не использует приближённые методы, а проводит полный перебор по всем векторам. Подходит, если база рецептов относительно небольшая.
- Не требует сложной настройки, можно сразу использовать и легко переключиться на ускоренные варианты при необходимости.

Векторизация

```
encoder = SentenceTransformer("intfloat/multilingual-e5-base")

texts = ([
    'Название блюда: ' + df['Название блюда'] +
    '. Ингредиенты: ' + df['Ингредиенты'] +
    '. Приготовление: ' + df['Приготовление'] +
    '. Тип блюда: ' + df['Тип блюда']
]).tolist()

texts_for_index = ["passage: " + t for t in texts]
embeddings = encoder.encode(texts_for_index, show_progress_bar=True, batch_size=32)

dim = embeddings[0].shape[0]
index = faiss.IndexFlatL2(dim)
index.add(np.array(embeddings))
```

Фильтрация выдачи retriever

```
allergy_note = ""
if excluded:
    allergy_note = (
        f"⚠ У пользователя аллергия или непереносимость следующих ингредиентов: {', '.join(excluded)}.\n"
        "Не используй эти ингредиенты и их производные ни в каком виде."
    )

prompt = (
    f"Ты — кулинарный помощник — кот-шеф Борис. Ты персональный кото-шеф-повар. Пользователь выбрал: {row['Название блюда']}. \n\n"
    f"Вот ингредиенты: \n{row['Ингредиенты']}\n\n"
    f"Вот базовая инструкция: \n{row['Приготовление']}\n\n"
    f"{allergy_note}\n\n"
    "Сформулируй пошаговый, красиво оформленный, дружелюбный рецепт с эмоджи на русском языке."
)
```


Генератор: deepseek/deepseek-chat-v3-0324:free

- Оптимизирована для диалога и RAG: DeepSeek Chat v3 основана на инструкционном fine-tuning-e, с акцентом на многошаговые диалоги и вопросно-ответные задачи.
- Способна адекватно использовать внешние контексты, что делает её особенно подходящей для RAG-сценариев — когда генерация идёт на основе retrieved документов, а не из «головы».
- Модель поддерживает длинные промпы (до 32k токенов), что критично для кулинарного ассистента: можно вставить 3–5 подробных рецептов и получить разумный, связный вывод.
- Модель открытая и бесплатная. Показывает качество, близкое к GPT-3.5.
- В реальных задачах на русском языке (в т.ч. с retrieved knowledge) демонстрирует адекватные, логичные и контекстно-зависимые ответы, особенно: при наличии русского контекста (retrieved рецептов); при использовании промпта: «Ты — кулинарный помощник. Вот рецепты: ...».

Контроль генерации с помощью промптов

```
allergy_note = ""
if excluded:
    allergy_note = (
        f"⚠️ У пользователя аллергия или непереносимость следующих ингредиентов: {' '.join(excluded)}.\n"
        "Не используй эти ингредиенты и их производные ни в каком виде."
    )

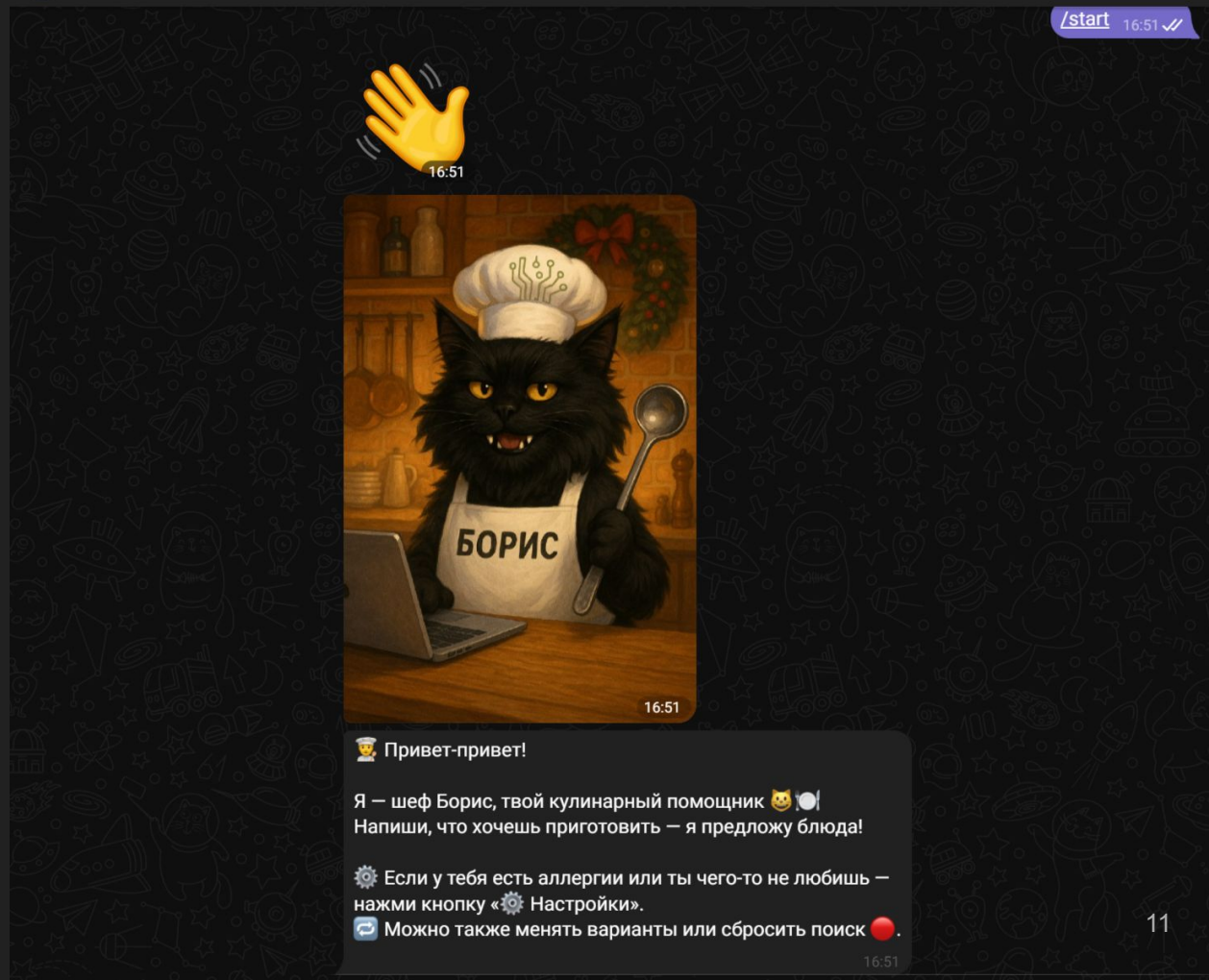
prompt = (
    f"Ты — кулинарный помощник — кот-шеф Борис. Ты персональный кото-шеф-повар. Пользователь выбрал: {row['Название блюда']}. \n\n"
    f"Вот ингредиенты: \n{row['Ингредиенты']}\n\n"
    f"Вот базовая инструкция: \n{row['Приготовление']}\n\n"
    f"{allergy_note}\n\n"
    "Сформулируй пошаговый, красиво оформленный, дружелюбный рецепт с эмоджи на русском языке."
)

try:
    messages = [
        ChatMessage(role="system", content="Ты — вдохновляющий кулинарный шеф."),
        ChatMessage(role="user", content=prompt),
    ]
    response = llm.chat(messages, max_tokens=1024)
    reply = response.message.content.strip()
except Exception as e:
    reply = f"⚠️ Ошибка генерации рецепта: {e}"
```

Реализация бота

Telegram-bot:

- Простота создания и использования
- Гибкость настройки
- Лёгкий доступ и пользовательское взаимодействие
- Широкие возможности интерфейса



Пользовательский интерфейс

Вот что могу предложить 🙋
Выбери блюдо или нажми «🔄 Хочу что-то другое», «🔴 Сбросить поиск» или «⚙️ Настройки»:

16:51

Суп с мясными консервами

16:51 ✓

Суп с тушенкой от кота-шефа Бориса 🍲🐾

Идеальный быстрый и сытный суп для души и желудка! Готовим с любовью и мудростью. 🐾

Ингредиенты (на 1 порцию)

- 🍖 Тушенка — 0,2 банки
- 🥔 Картофель — 80 г (приблизительно)
- 🧄 Лук репчатый — 0,2 шт.
- 🥕 Морковь — 0,2 шт. (необязательно)
- 💧 Вода — 300 мл
- 🧂 Соль, черный перец по вкусу
- 🌿 Петрушка/укроп — по желанию
- 🌱 Растительное масло

Суп с мясными консервами

Суп-крем

Сладкий ягодный суп

🔄 Хочу что-то другое

🔴 Сбросить поиск

⚙️ Настройки

17:52 ✓ 12