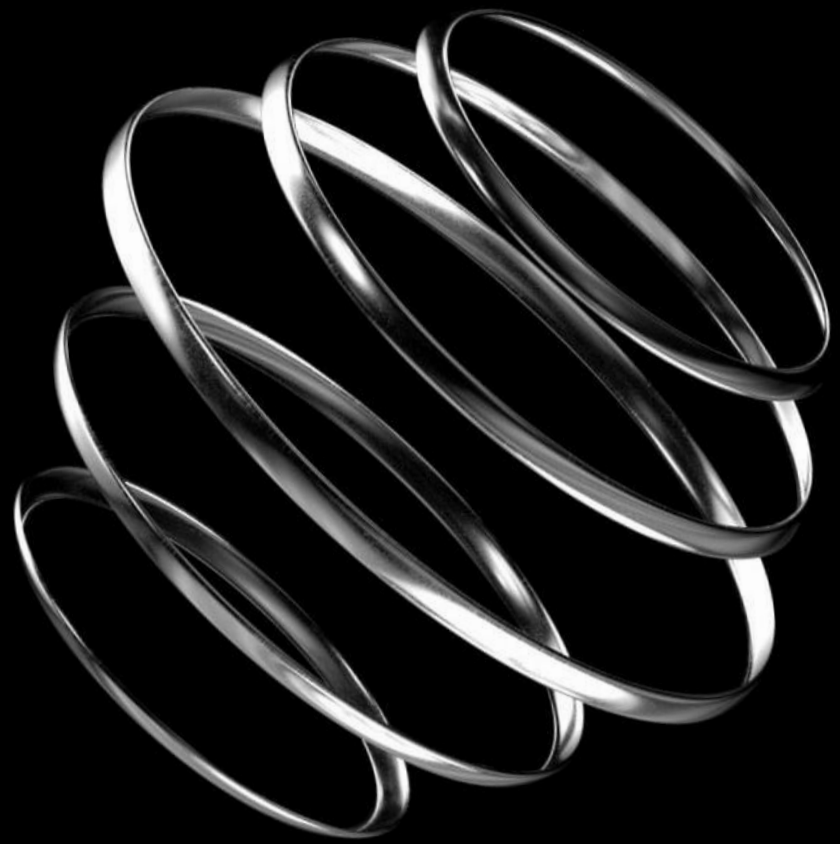
Тестирование LLM приложений с DeepEval



© red_mad_robot

О себе

- NLP-инженер в red_mad_robot. Разрабатываю, улучшаю и исследую агентные системы;
- С 2022 года занимаюсь AI. Участвовал в работе над проектами, связанными с Classic ML, Computer Vision, NLP, RL.

Максимов Максим

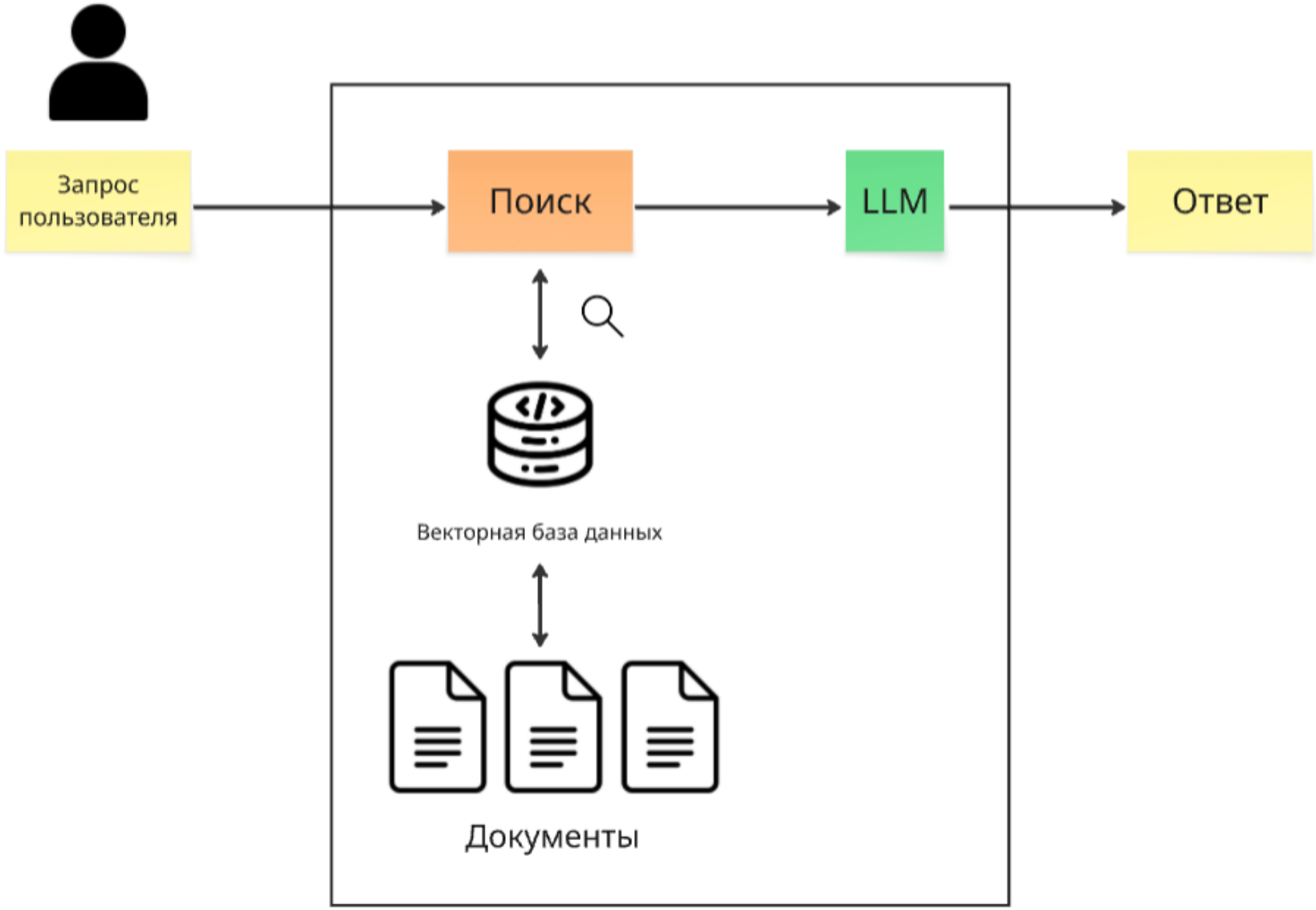
© red_mad_robot

Представим, у нас есть LLM ассистент с RAG по продуктовой базе супермаркета.

Он помогает пользователям находить нужные товары, а также отвечает на вопросы по ним.

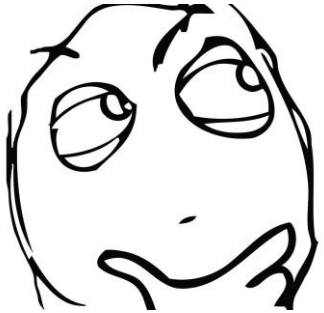
© red_mad_robot

LLM ассистент с RAG по продуктовой базе



© red_mad_robot,

Как протестировать нашего ассистента?



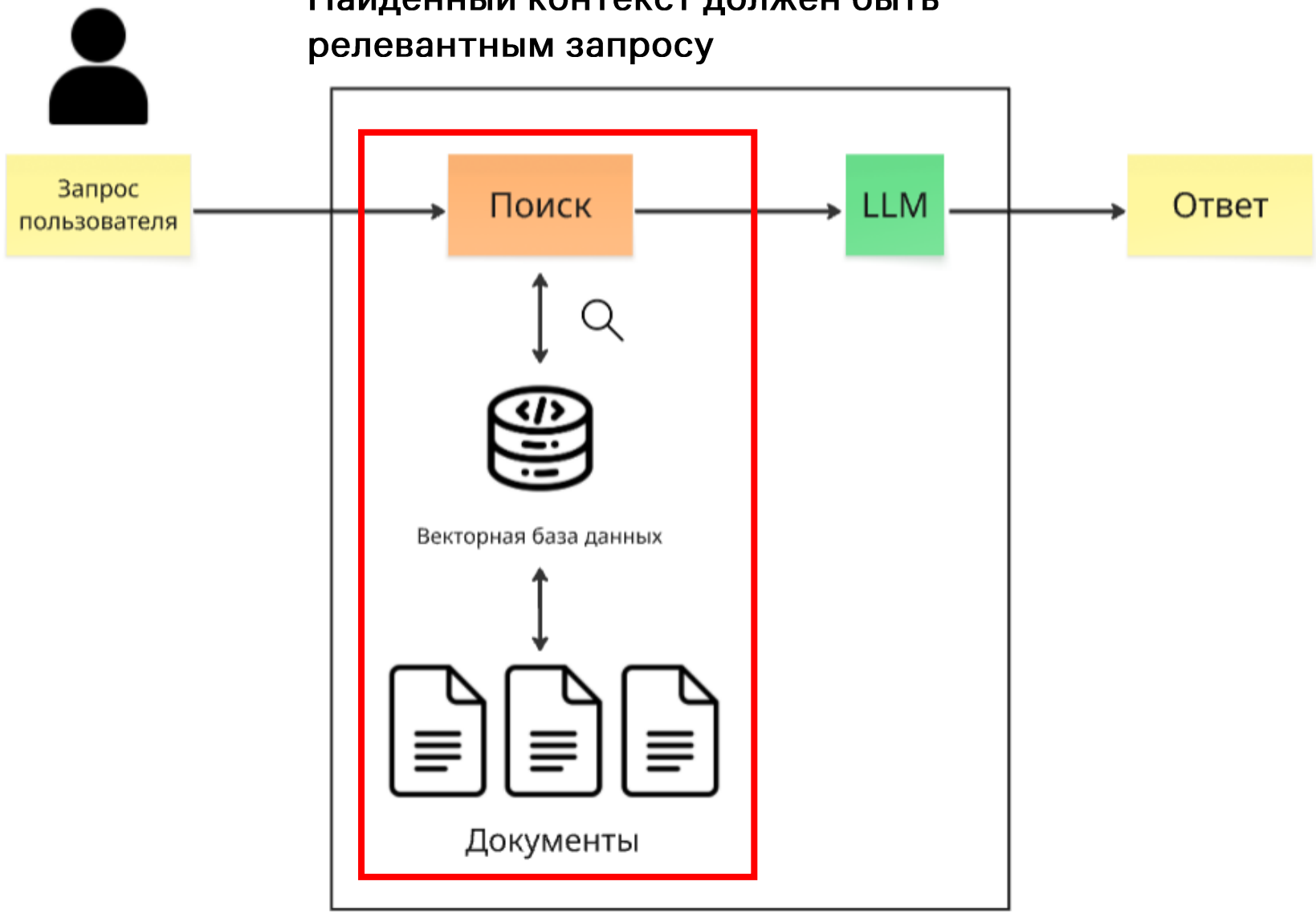
© red_mad_robot,

Наверное, классическое Unit-тестирование!

© red_mad_robot

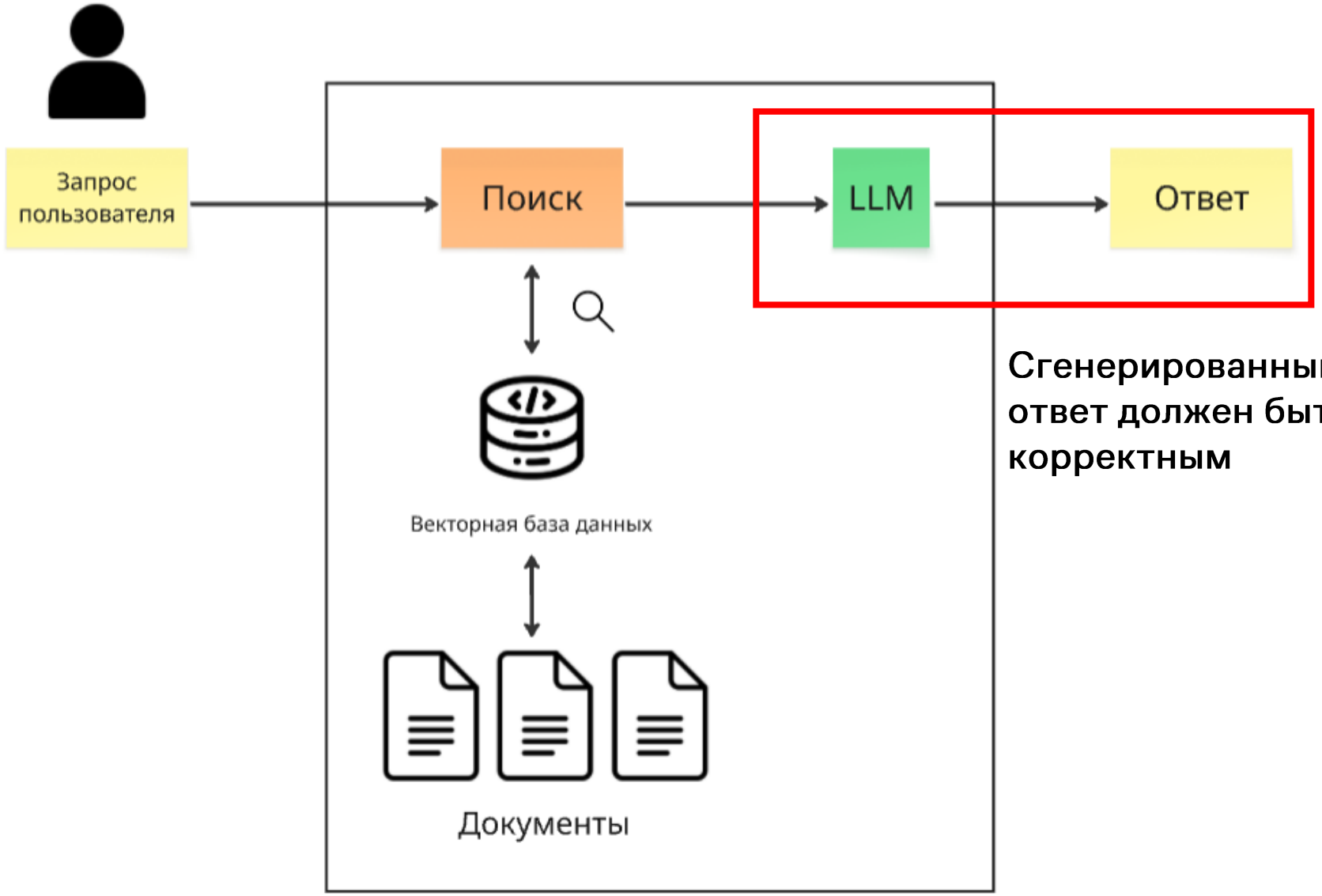
Не все так просто...

Найденный контекст должен быть релевантным запросу



© red_mad_robot

Не все так просто...



Сгенерированный ответ должен быть корректным

© red_mad_robot

Тестирование LLM приложений

© red_mad_robot

Почему тестирование LLM – это сложно

Недетерминированность

Работа с естественным языком

© red_mad_robot,

Попробуем протестировать и оценить работу нашего ассистента

© red_mad_robot

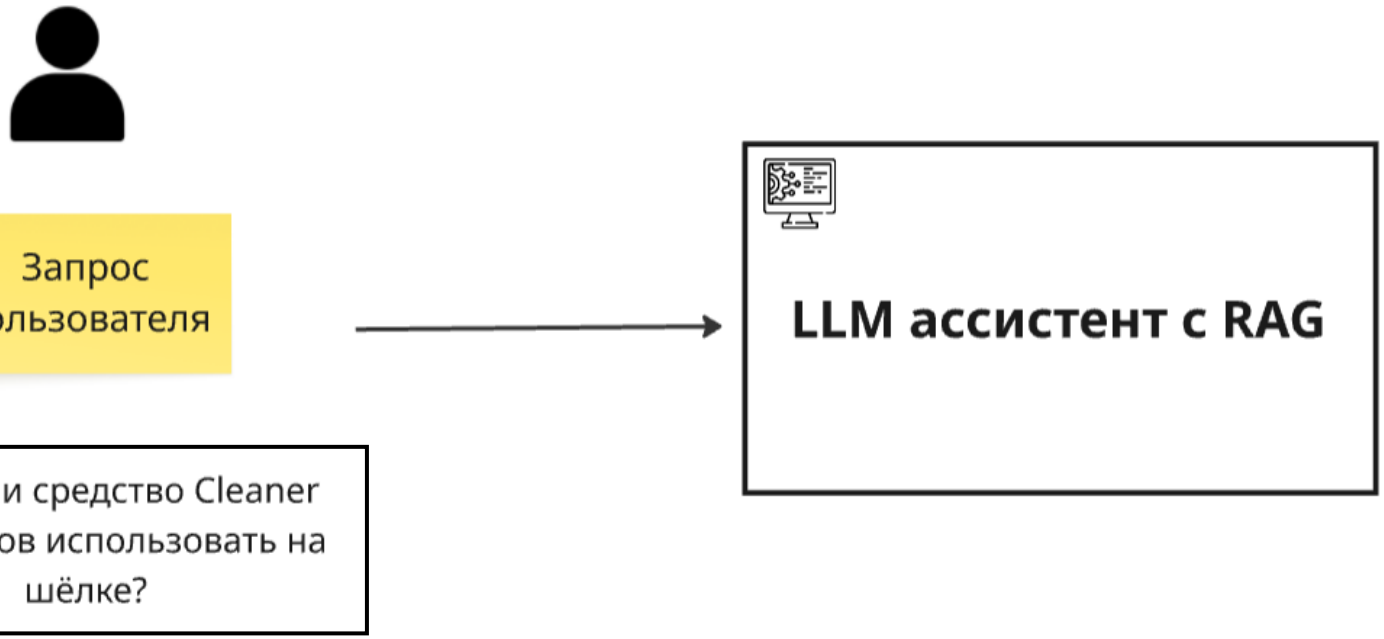
Работа LLM ассистента с RAG

Запрос пользователя

Можно ли средство Cleaner для ковров использовать на шёлке?

© red_mad_robot,

Работа LLM ассистента с RAG



© red_mad_robot,

Работа LLM ассистента с RAG

Запрос пользователя

Можно ли средство Cleaner для ковров использовать на шёлке?

LLM ассистент с RAG

Ответ системы

Cleaner можно применять на шёлке

© red_mad_robot,

Работа LLM ассистента с RAG

Запрос пользователя

Можно ли средство Cleaner для ковров использовать на шёлке?

Ответ системы

Cleaner можно применять на шёлке

Правильный ответ

Cleaner нельзя применять на шёлке

© red_mad_robot,

Классические NLP метрики

BLUE

ROUGE

METEOR

© red_mad_robot

Классические NLP метрики

BLUE

ROUGE

METEOR

*они здесь не столь релевантны, посмотрим почему

© red_mad_robot

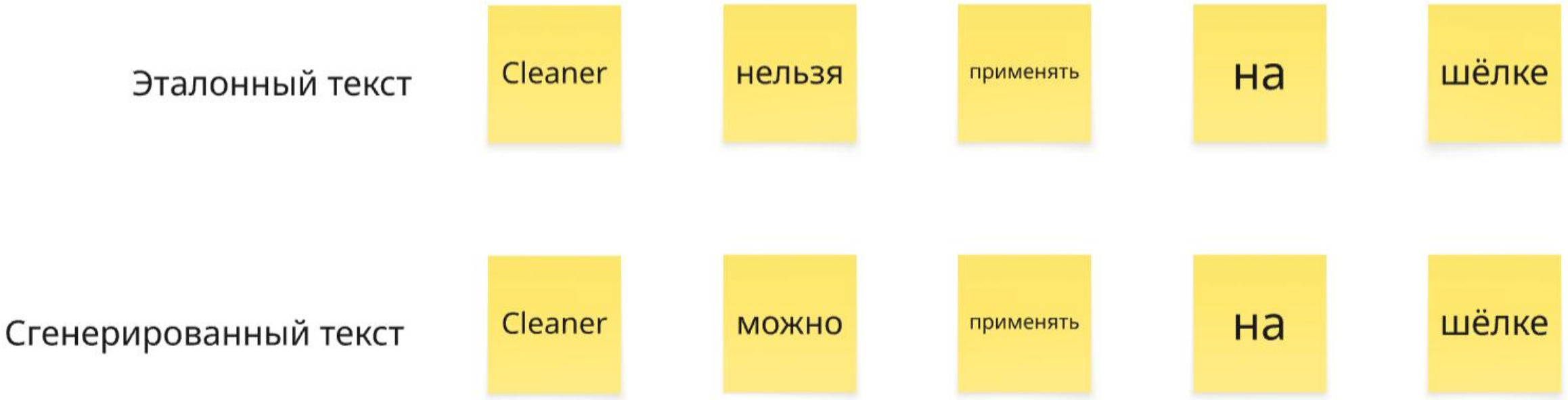
В качестве примера, возьмем ROUGE

© red_mad_robot

Подсчет метрики ROUGE

© red_mad_robot,

Подсчет метрики ROUGE



© red_mad_robot,

Подсчет метрики ROUGE

Находим пересечение токенов

© red_mad_robot,

Подсчет метрики ROUGE

$$R = \{Cleaner,\ нельзя,\ применять,\ на,\ шёлке\}, \quad |R| = 5$$

$$C = \{Cleaner,\ можно,\ применять,\ на,\ шёлке\}, \quad |C| = 5$$

$$R \cap C = \{Cleaner,\ применять,\ на,\ шёлке\}, \quad |R \cap C| = 4$$

© red_mad_robot,

Подсчет метрики ROUGE

$$\text{Recall} = \frac{|R \cap C|}{|R|} = \frac{4}{5} = 0.8$$

$$\text{Precision} = \frac{|R \cap C|}{|C|} = \frac{4}{5} = 0.8$$

$$F_1 = \text{ROUGE-1} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \cdot \frac{0.8 \times 0.8}{0.8 + 0.8} = 0.8$$

© red_mad_robot,

Подсчет метрики ROUGE

$$\text{Recall} = \frac{|R \cap C|}{|R|} = \frac{4}{5} = 0.8$$

$$\text{Precision} = \frac{|R \cap C|}{|C|} = \frac{4}{5} = 0.8$$

$$F_1 = \text{ROUGE-1} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = 2 \cdot \frac{0.8 \times 0.8}{0.8 + 0.8} = \boxed{0.8}$$

Высокий score!

© red_mad_robot,

Подсчет метрики ROUGE

© red_mad_robot,

Вывод: Классические метрики считают совпадения слов, а не смысл.

© red_mad_robot

Решение – LLM-as-a-Judge

© red_mad_robot

LLM-as-a-Judge

Запрос пользователя

Можно ли средство Cleaner для ковров использовать на шёлке?

LLM ассистент с RAG

Ответ системы

Cleaner можно применять на шёлке

Правильный ответ

Cleaner нельзя применять на шёлке

© red_mad_robot,

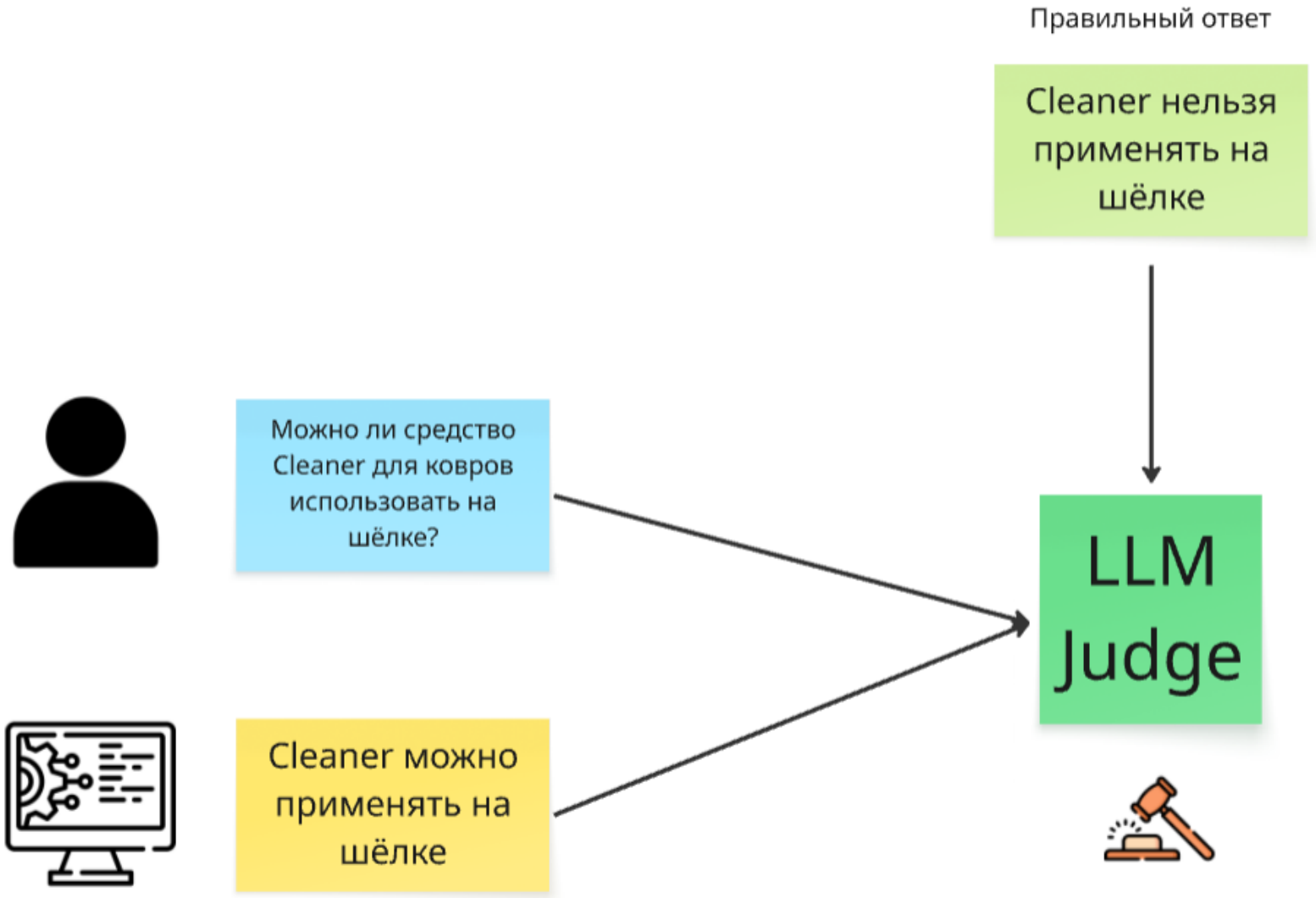
LLM-as-a-Judge

Можно ли средство Cleaner для ковров использовать на шёлке?

Cleaner можно применять на шёлке

© red_mad_robot,

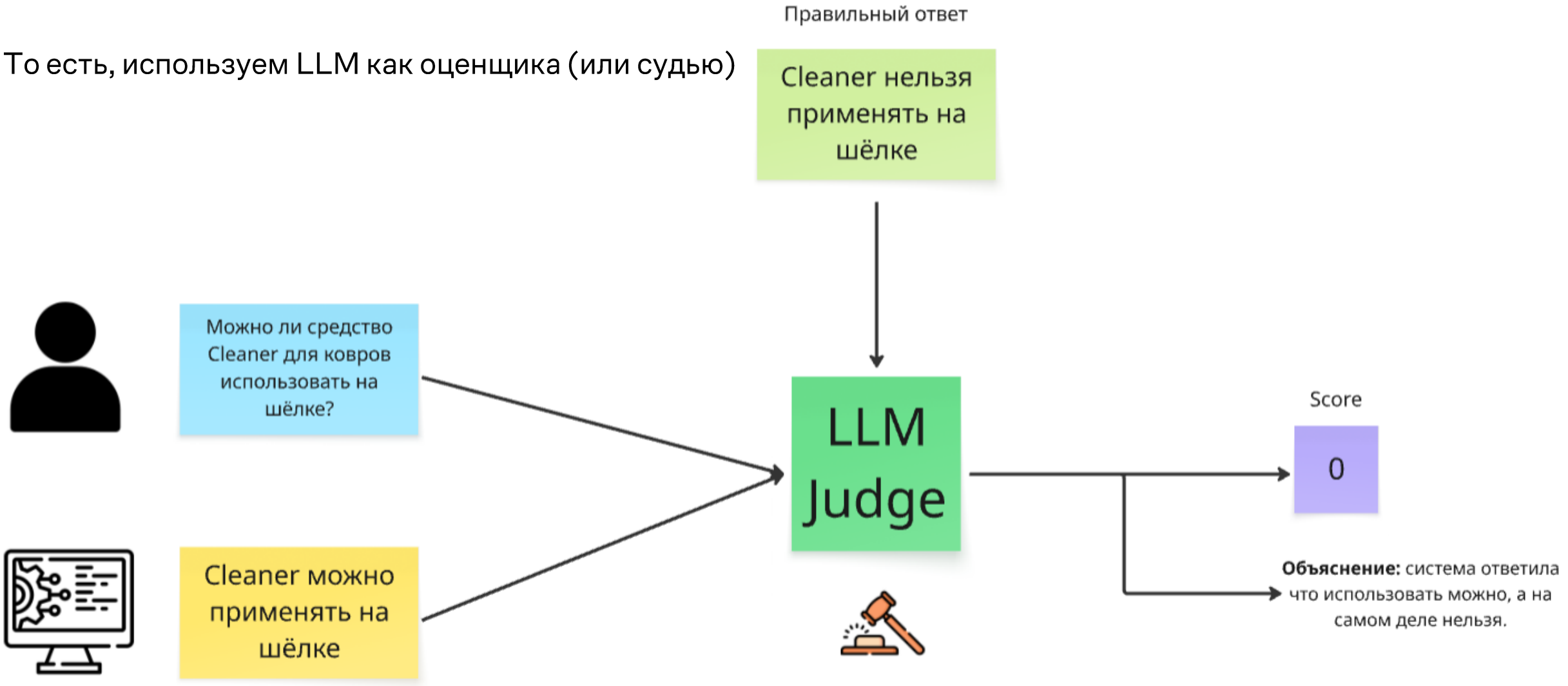
LLM-as-a-Judge



© red_mad_robot,

LLM-as-a-Judge

То есть, используем LLM как оценщика (или судью)



© red_mad_robot,

То есть, просто берем LLM, промпты и тестируем?

© red_mad_robot

На самом деле нет...

Хочется сделать следующее:

1. Не только подсчитать метрики, но и сравнить различные подходы
2. Не реализовывать с нуля алгоритмы оценки, а использовать готовые
3. Внедрить и произвести оценку в существующую инфраструктуру

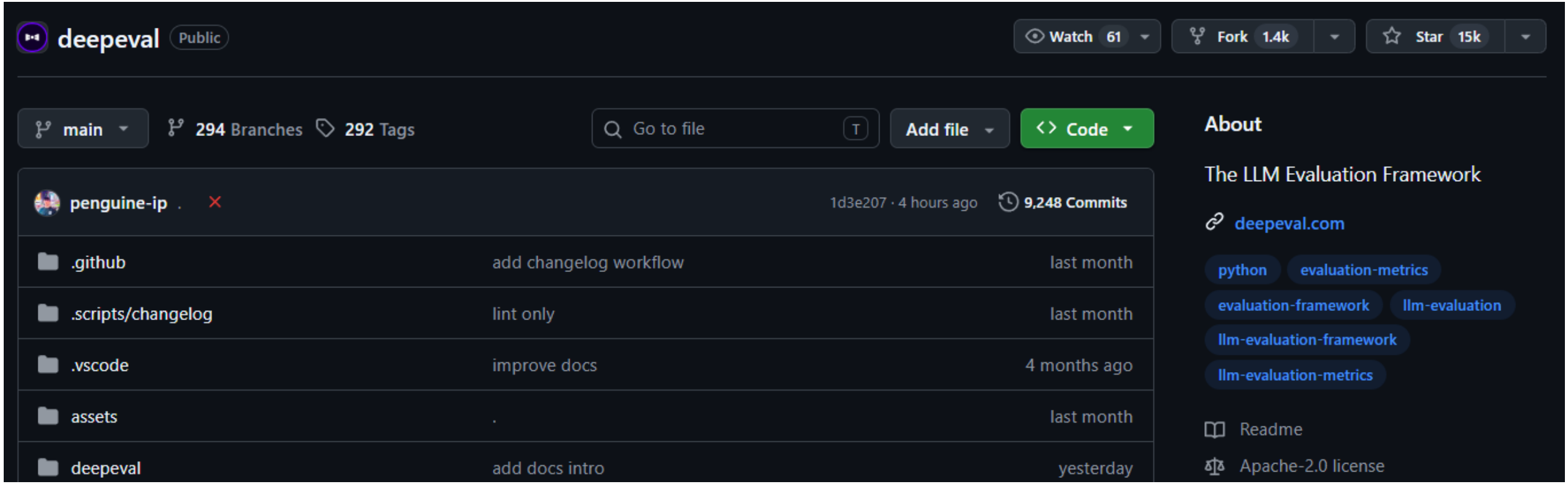
© red_mad_robot,

Что такое DeepEval

© red_mad_robot

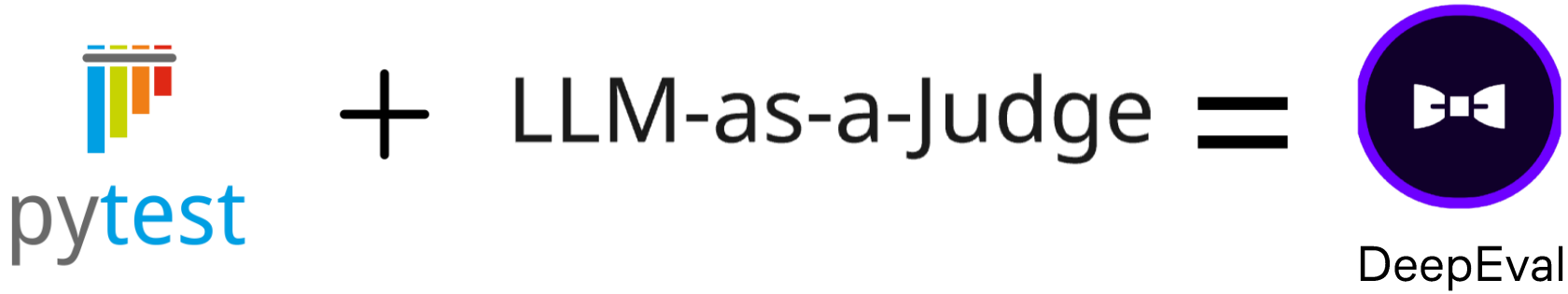
Что такое DeepEval

DeepEval – это open-source платформа для оценки и тестирования приложений, работающих на основе LLM



© red_mad_robot,

Что такое DeepEval



© red_mad_robot,

Сферы применения DeepEval

Чат боты

RAG

AI-агенты

LLM арена

© red_mad_robot,

Сферы применения DeepEval

Чат боты

RAG

AI-агенты

Подходит!

LLM арена

© red_mad_robot,

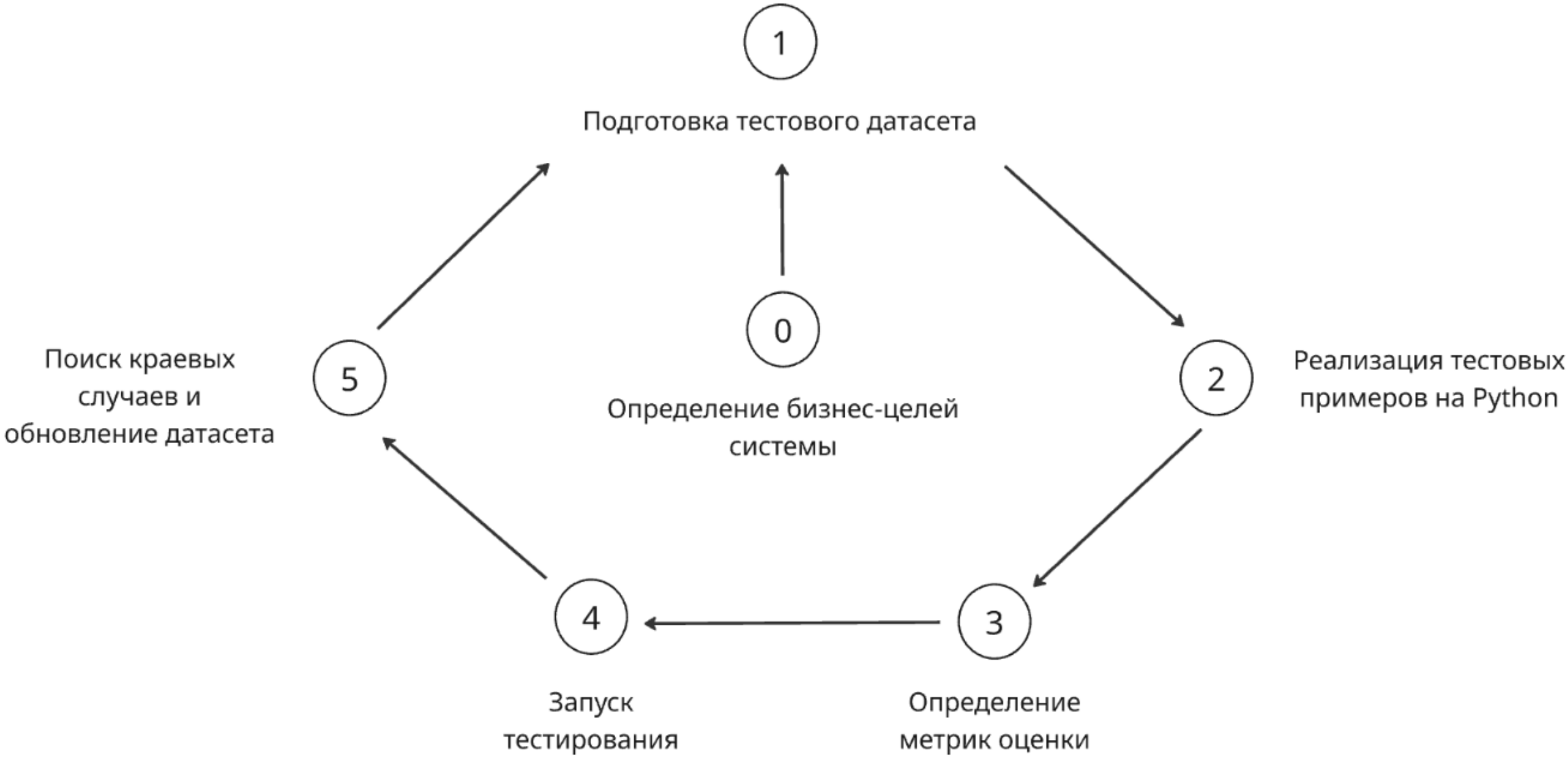
С чего начать?

© red_mad_robot

Процесс тестирования с DeepEval

© red_mad_robot,

Процесс тестирования с DeepEval



© red_mad_robot,

Протестируем нашего ассистента с DeepEval

© red_mad_robot

Процесс тестирования с DeepEval

© red_mad_robot,

Определяем бизнес-цель ассистента

Повышение лояльности клиентов

© red_mad_robot,

Определяем бизнес-цель ассистента

Повышение лояльности клиентов → Нужно чтобы ассистент давал точные ответы на вопросы

© red_mad_robot,

Процесс тестирования с DeepEval

© red_mad_robot,

Форматы работы с датасетами в DeepEval

© red_mad_robot

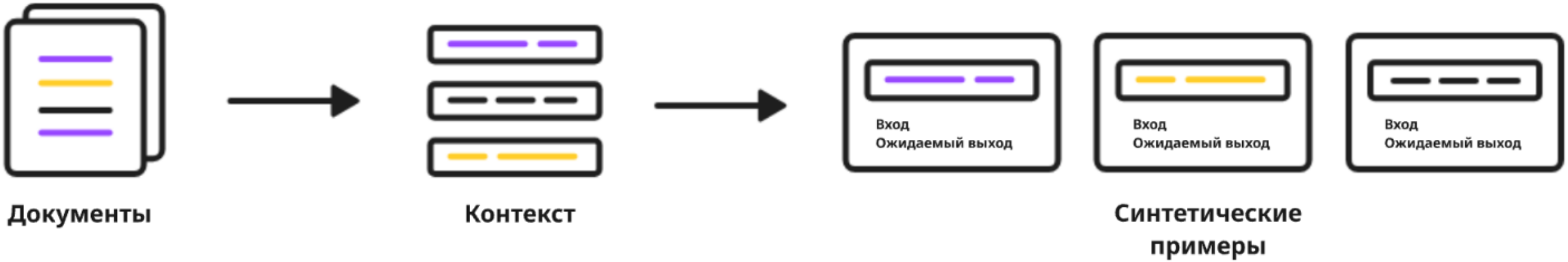
Тестовый датасет:

Эталонный набор «вопрос — ответ» для оценки качества LLM-ассистента.

© red_mad_robot

Работа с датасетами в DeepEval

Генерация датасета из документов



© red_mad_robot,

Работа с датасетами в DeepEval

Используем если:

- Имеются документы в форматах pdf, docx, txt

Генерация датасета из документов

© red_mad_robot,

Работа с датасетами в DeepEval

Генерация датасета из сформированной базы данных

© red_mad_robot,

Работа с датасетами в DeepEval

Используем если:

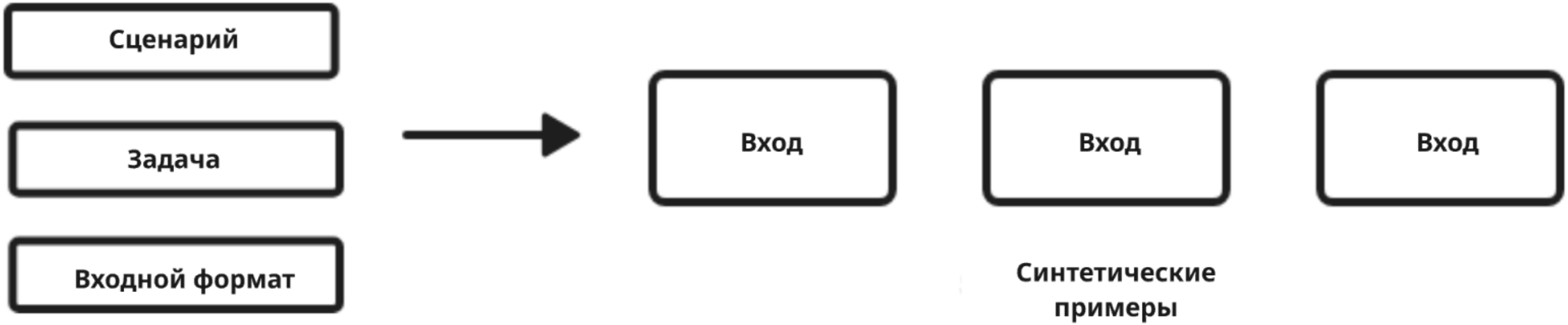
- Собрана векторная база данных с фрагментами контекста

Генерация датасета из сформированной базы данных

© red_mad_robot,

Работа с датасетами в DeepEval

Генерация датасета с нуля по собственному сценарию



© red_mad_robot,

Работа с датасетами в DeepEval

Генерация датасета с нуля по собственному сценарию

Используем если:

- Необходимо создать датасет, выходящий за рамки существующей базы знаний с нуля

© red_mad_robot,

Подготовим тестовый датасет

© red_mad_robot

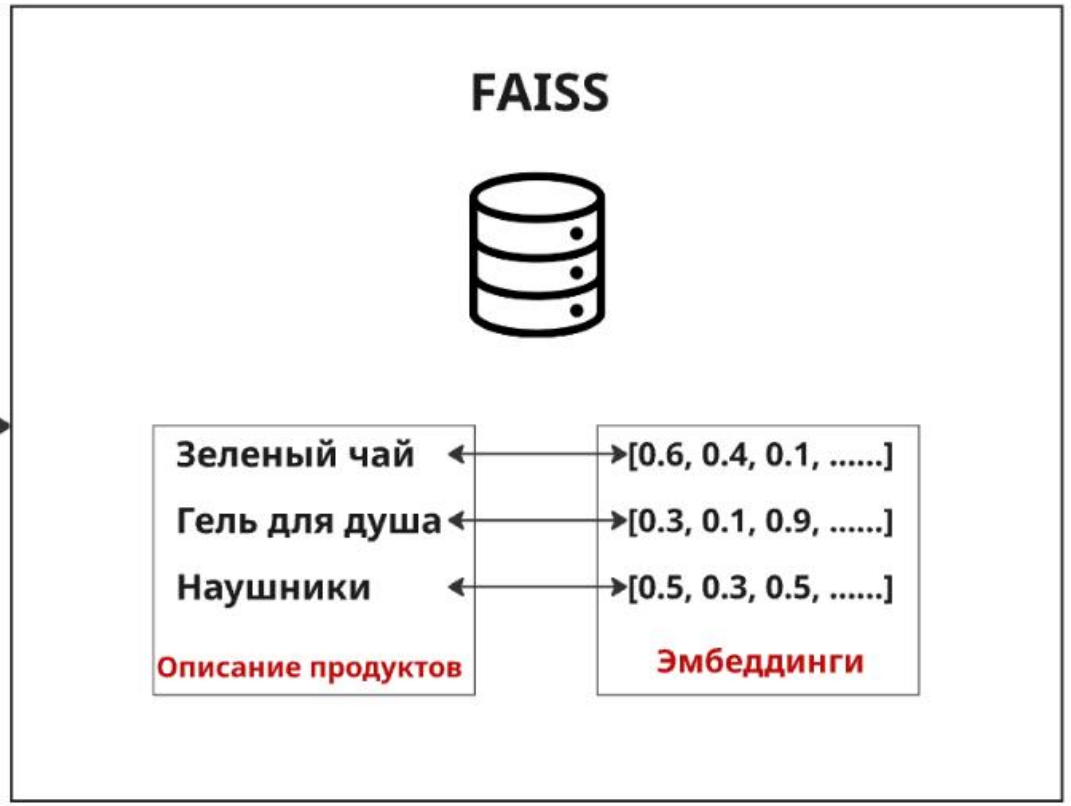
Подготовка тестового датасета

products.txt

Файл Изменить Просмотр

```
1. Органический зелёный чай «Утренняя роса»

Описание: Нежный чай высшего сорта, собранный на высокогорных
плантациях. Имеет лёгкий травянистый аромат и мягкий, слегка
сладковатый вкус с долгим послевкусием. Идеален для начала дня.

Цена: 450 руб.

Состав: лист зелёного чая (100%).

2. Умная колонка «SoundBox Mini»

Описание: Компактный умный динамик с голосовым помощником
Алисой. Управляет умным домом, воспроизводит музыку с потоковых
сервисов, отвечает на вопросы. Качество звука сбалансированное,
с чёткими высокими частотами.

Цена: 3290 руб.

Состав: пластиковый корпус, электронные компоненты (процессор,
модули Wi-Fi/Bluetooth, динамик), резиновое основание.
```

Строка 6, столбец 1 | 35 437 символов | Обычный текст | 190% | Windows (CRLF) | UTF-8



© red_mad_robot,

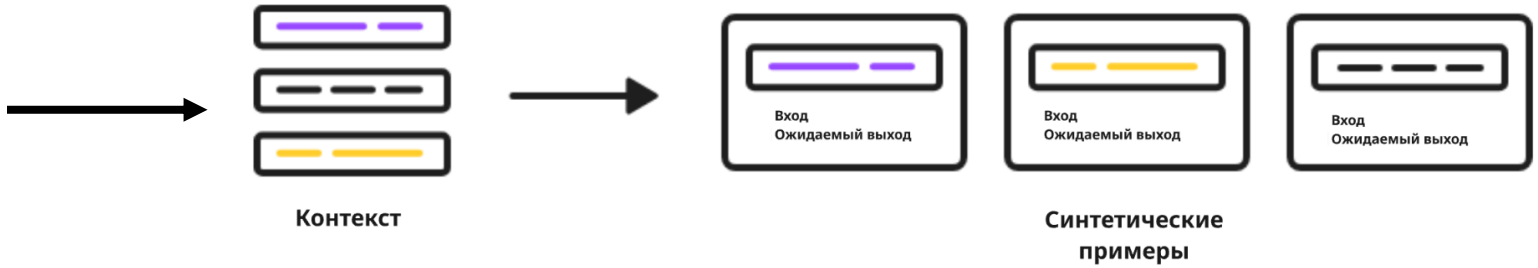
Подготовка тестового датасета

products.txt

```
1. Органический зелёный чай «Утренняя роса»
Описание: ...
Цена: 450 руб.

2. Умная колонка «SoundBox Mini»
Описание: ...
Цена: 3290 руб.

3. Гель для душа «Морская свежесть»
Описание: ...
Цена: 280 руб.

...
```

FAISS

Генерация датасета из сформированной базы данных



© red_mad_robot,

```python
import ... # deepeval, data
...
model = CustomOpenAIModel(...)
contexts = [[chunk] for chunk in chunks]

synthesizer = Synthesizer(
    model=model,
    styling_config=StylingConfig(
        scenario="Покупатель интернет-магазина задаёт вопросы консультанту о
товарах. Все вопросы и ответы строго на русском языке.",
        task="Сгенерируй вопрос покупателя и ответ консультанта на русском языке",
    ),
)
goldens = synthesizer.generate_goldens_from_contexts(
    contexts=random.choices(contexts, k=10),
    include_expected_output=True,
    max_goldens_per_context=2,
)
...
df = pd.DataFrame(rows)
df.to_excel(...)
```

```python
import ... # deepeval, data
...
model = CustomOpenAIModel(...)
contexts = [[chunk] for chunk in chunks]

synthesizer = Synthesizer(
    model=model,
    styling_config=StylingConfig(
        scenario="Покупатель интернет-магазина задаёт вопросы консультанту о
товарах. Все вопросы и ответы строго на русском языке.",
        task="Сгенерируй вопрос покупателя и ответ консультанта на русском языке",
    ),
)
goldens = synthesizer.generate_goldens_from_contexts(
    contexts=random.choices(contexts, k=10),
    include_expected_output=True,
    max_goldens_per_context=2,
)
...
df = pd.DataFrame(rows)
df.to_excel(...)
```

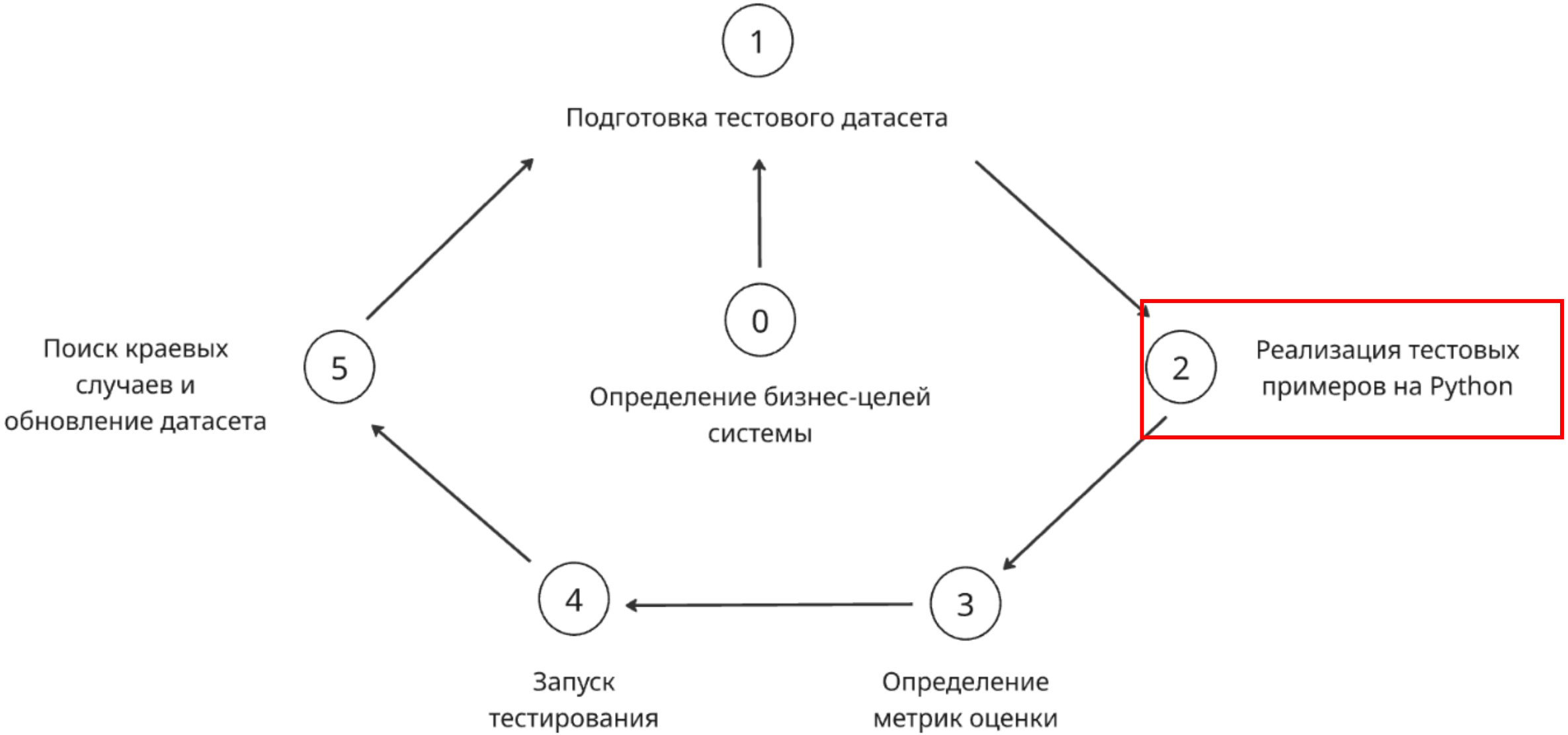
Подготовка тестового датасета

ВОПРОС (INPUT)	ОЖИДАЕМЫЙ ОТВЕТ (EXPECTED_OUTPUT)	КОНТЕКСТ (CONTEXT)
Сколько стоят наушники Listen Pro?	Наушники Listen Pro стоят **8 990 руб.**	товар Беспроводные наушники «Listen Pro». Цена: 8 990 руб.
Из чего состоит гель для душа Морская свежесть?	Содержит воду, ПАВ (кокамидопропилбетаин), экстракт морских водорослей, ментол, парфюмерную композицию и консервант.	состав Гель для душа «Морская свежесть». Состав: вода, ПАВ, экстракт морских водорослей, ментол, парфюмерная композиция, консервант. Цена: 280 руб.
Какой есть зеленый чай?	Имеется органический зелёный чай «Утренняя роса» за **450 руб.**	товар Органический зелёный чай «Утренняя роса». Цена: 450 руб. Нежный чай высшего сорта, собранный на высокогорных плантациях.

• • •

© red_mad_robot,

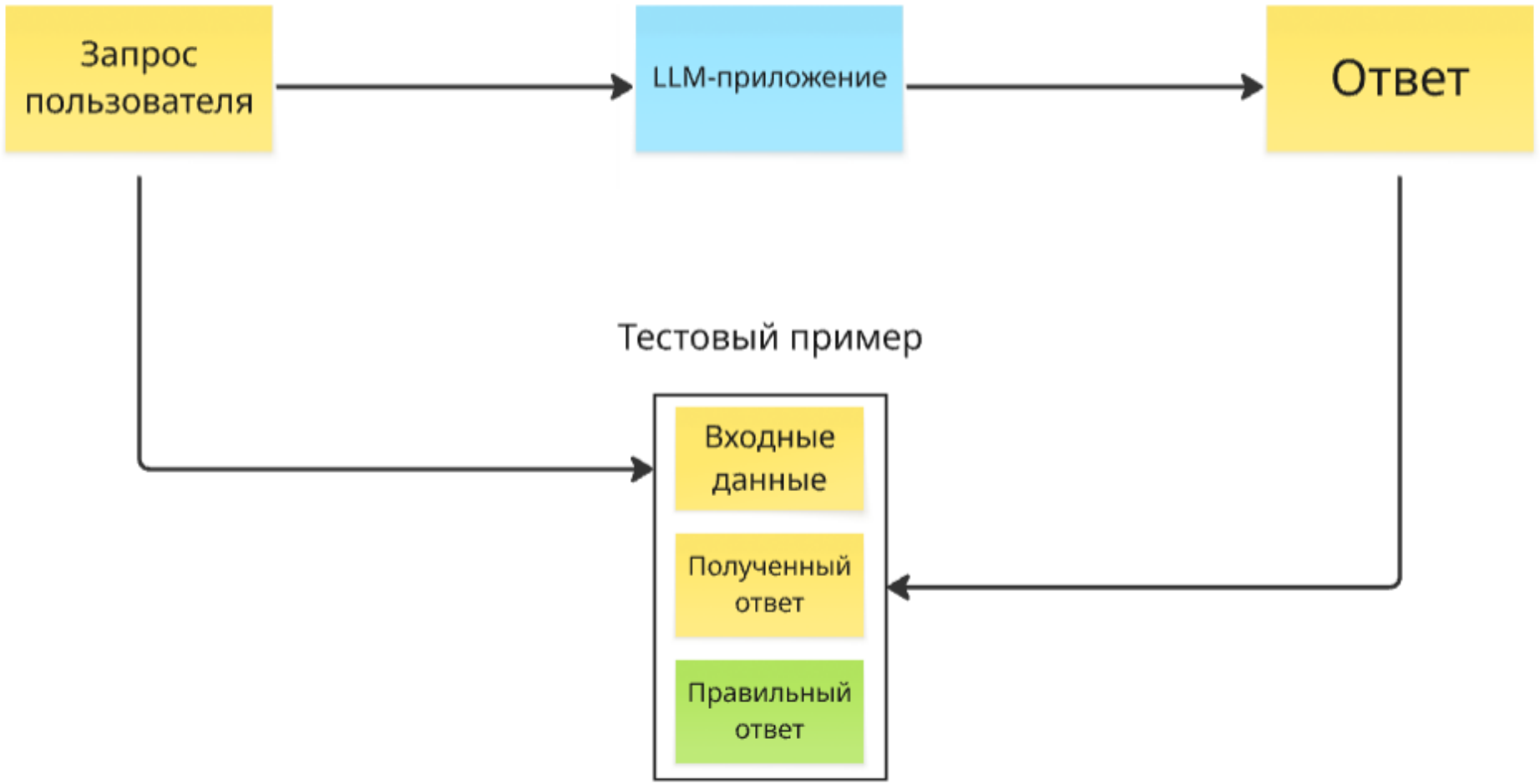
Процесс тестирования с DeepEval



© red_mad_robot,

Определение тестового примера

Атомарная единица взаимодействия с LLM-приложением



© red_mad_robot,

Типы тестовых примеров в DeepEval

Одношаговые

© red_mad_robot,

Одношаговые test case

Single-tern test case (одношаговый пример) – проверка одношаговой итерации взаимодействия с LLM-приложением

© red_mad_robot,

Типы тестовых примеров в DeepEval

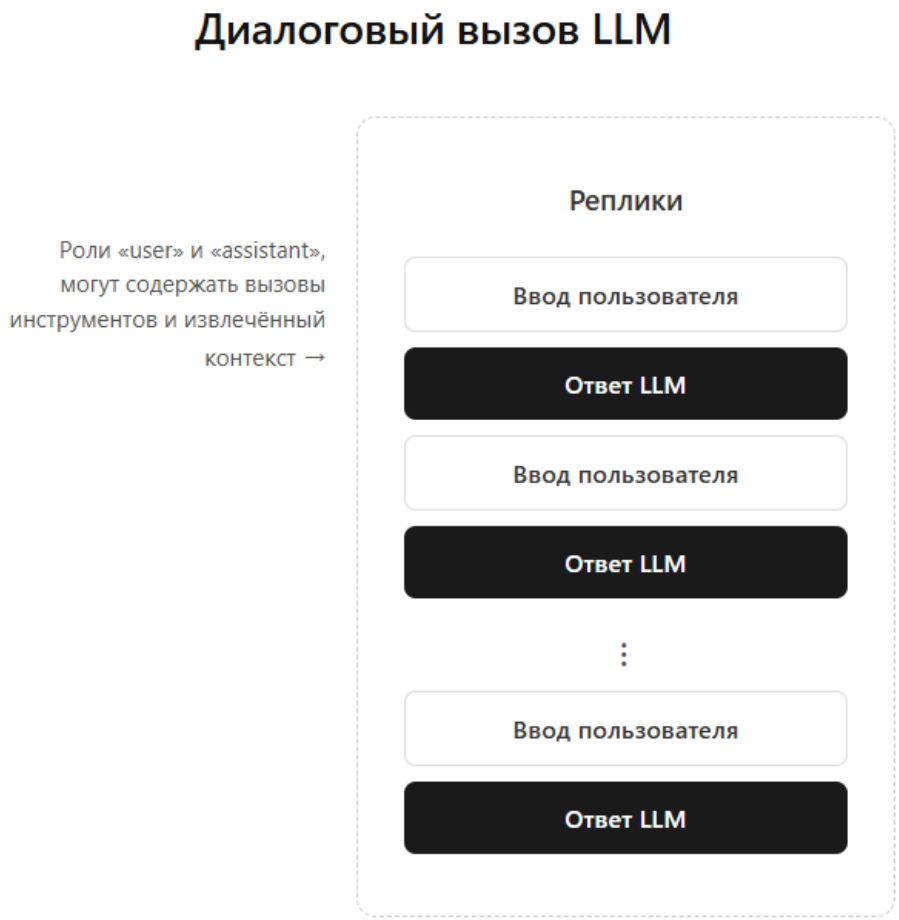
Одношаговые

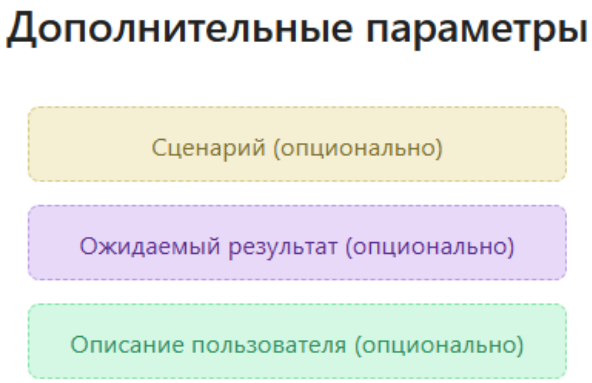
Многошаговые

© red_mad_robot,

Многошаговый test case

Multi-tern test case (многошаговый пример) – проверка серии взаимодействий с LLM-приложением





© red_mad_robot,

Типы тестовых примеров в DeepEval

Одношаговые

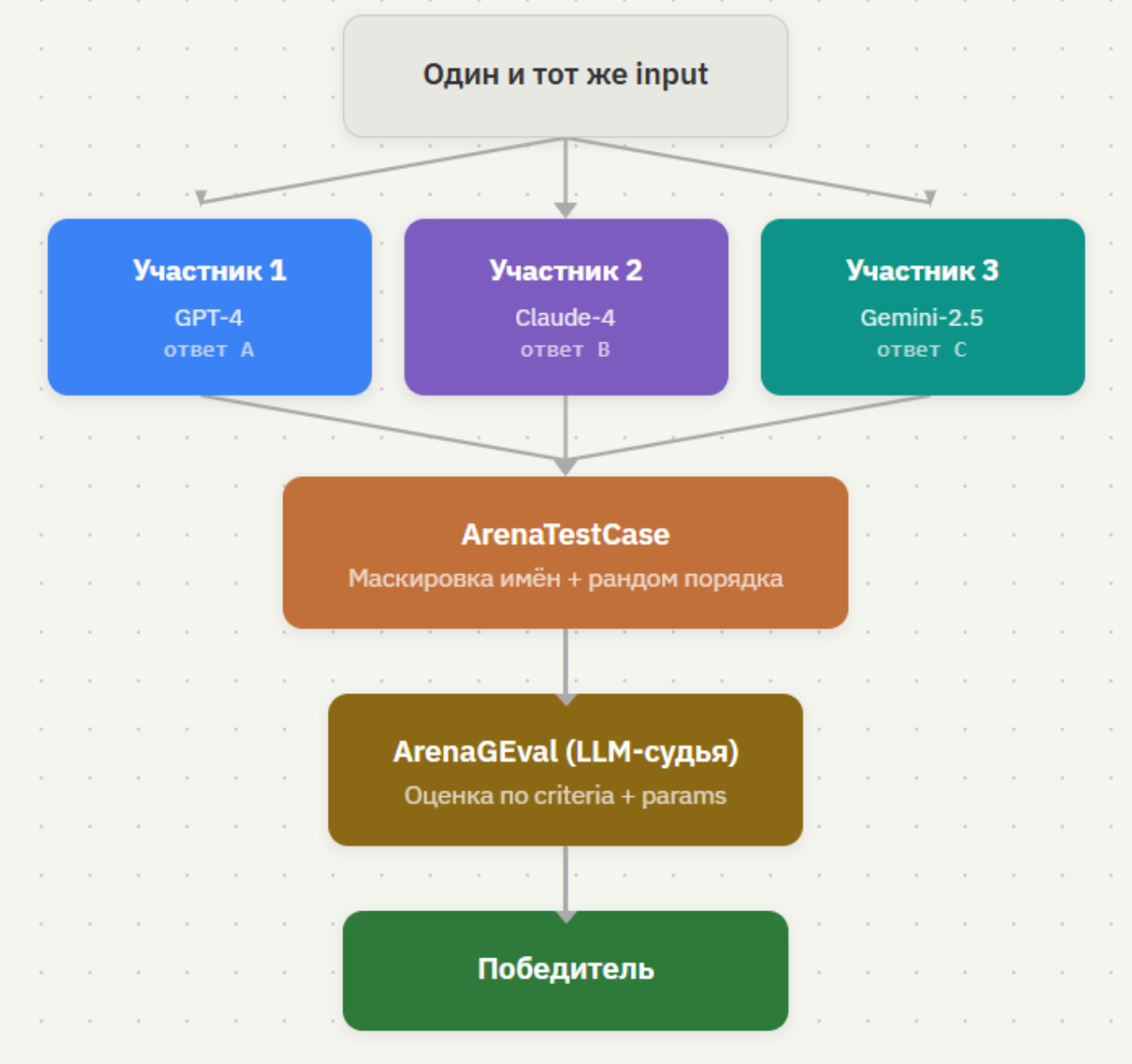
Многошаговые

Арена

© red_mad_robot,

Арена test case

Arena test case – сравнение различных LLM с различной конфигурациях на задачах.



© red_mad_robot,

Типы тестовых примеров в DeepEval

Одношаговые

Многошаговые

Арена

© red_mad_robot,

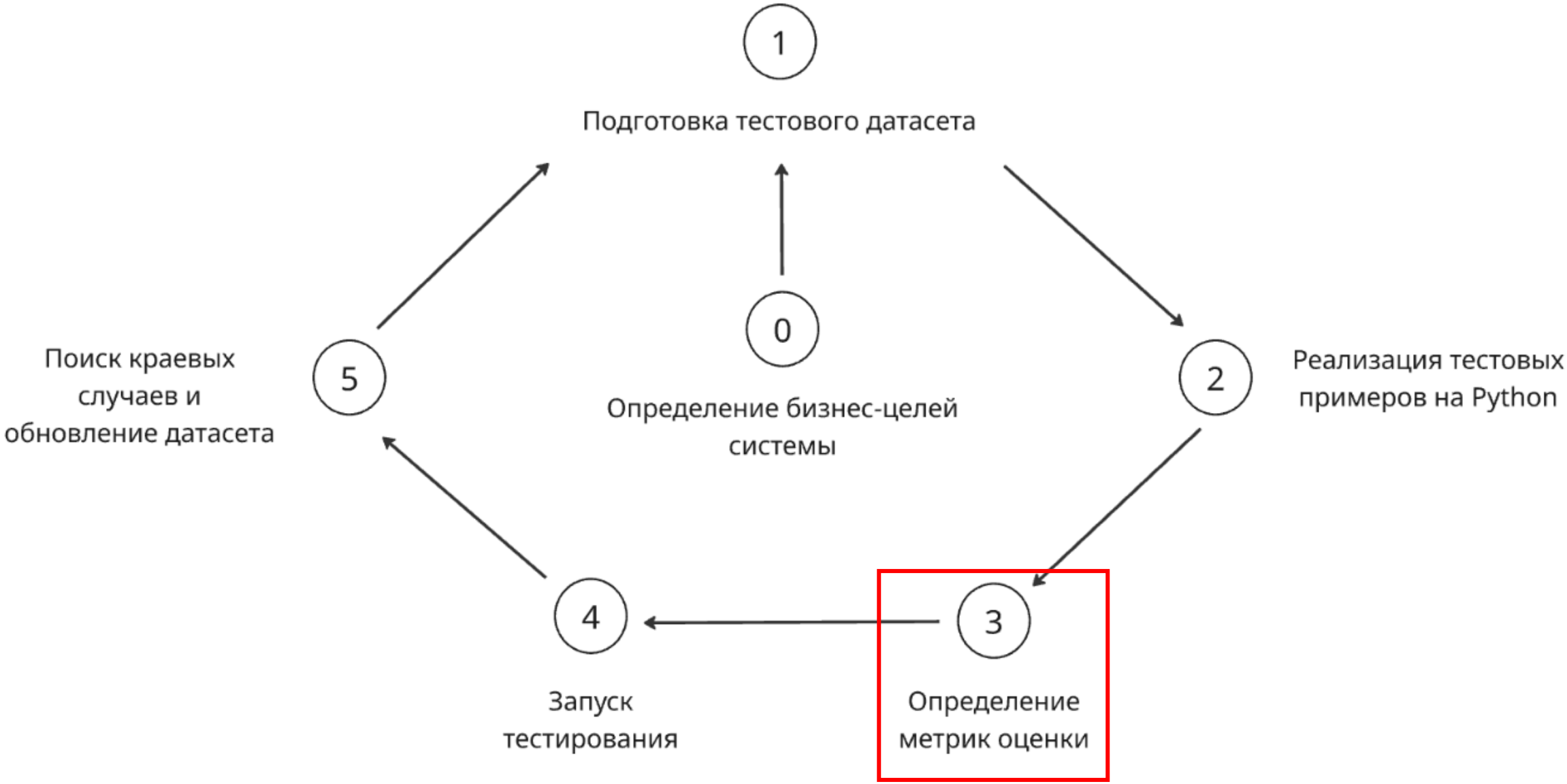
```python
from deepeval.test_case import LLMTestCase
from dataset import dataset
from rag_app import rag_pipeline

test_cases = []

for golden in dataset.goldens:
    # Прогоняем через RAG
    result = rag_pipeline(golden.input)

    test_case = LLMTestCase(
        input=golden.input,
        actual_output=result["actual_output"],
        expected_output=golden.expected_output,
        context=golden.context,  # ground truth контекст
        retrieval_context=result["retrieval_context"], # что реально нашёл RAG
    )
    test_cases.append(test_case)
```

©

```python
from deepeval.test_case import LLMTestCase
from dataset import dataset
from rag_app import rag_pipeline

test_cases = []

for golden in dataset.goldens:
    # Прогоняем через RAG
    result = rag_pipeline(golden.input)

    test_case = LLMTestCase(
        input=golden.input,
        actual_output=result["actual_output"],
        expected_output=golden.expected_output,
        context=golden.context,  # ground truth контекст
        retrieval_context=result["retrieval_context"], # что реально нашёл RAG
    )
    test_cases.append(test_case)
```

©

Процесс тестирования с DeepEval



© red_mad_robot,

Метрики в DeepEval

Метрики в DeepEval – данный инструмент содержит **более 50 готовых** к использованию метрик для различных задач

© red_mad_robot,

Метрики в DeepEval

Метрики в DeepEval – данный инструмент содержит **более 50 готовых** к использованию метрик для различных задач

© red_mad_robot,

Выберем метрики для оценки RAG

Answer Relevancy (релевантность ответа)

Оценка того, как хорошо наша система генерирует ответ.

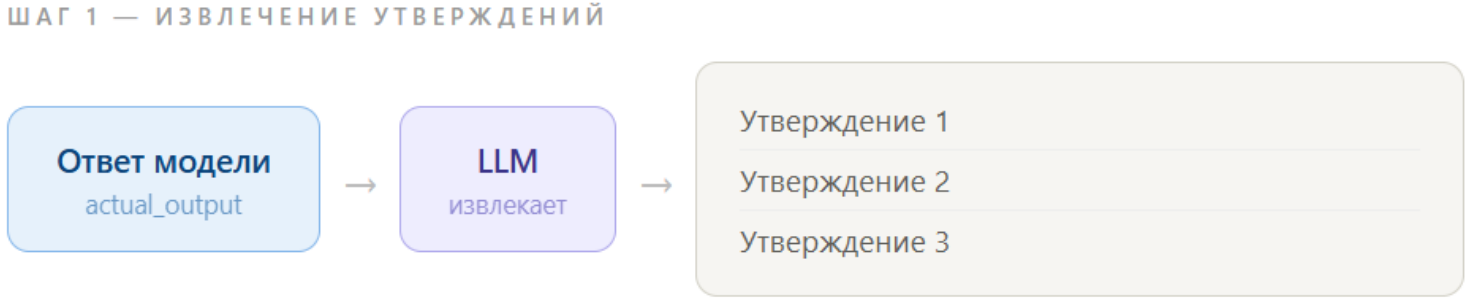
© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Answer Relevancy (релевантность ответа)

Оценка того, как хорошо наша система генерирует ответ.



© red_mad_robot,

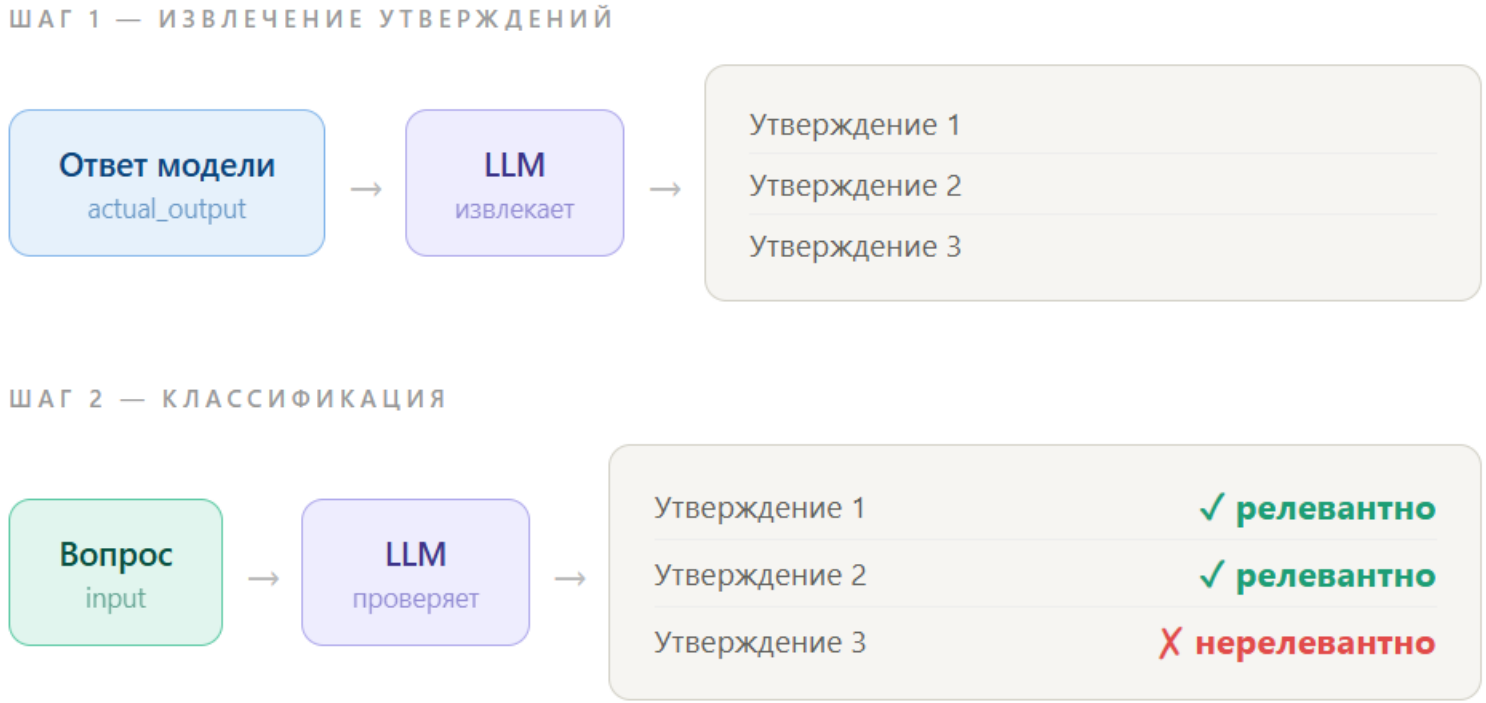
https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Возьмем 2 метрики:

Answer Relevancy (релевантность ответа)

Оценка того, как хорошо наша система генерирует ответ.



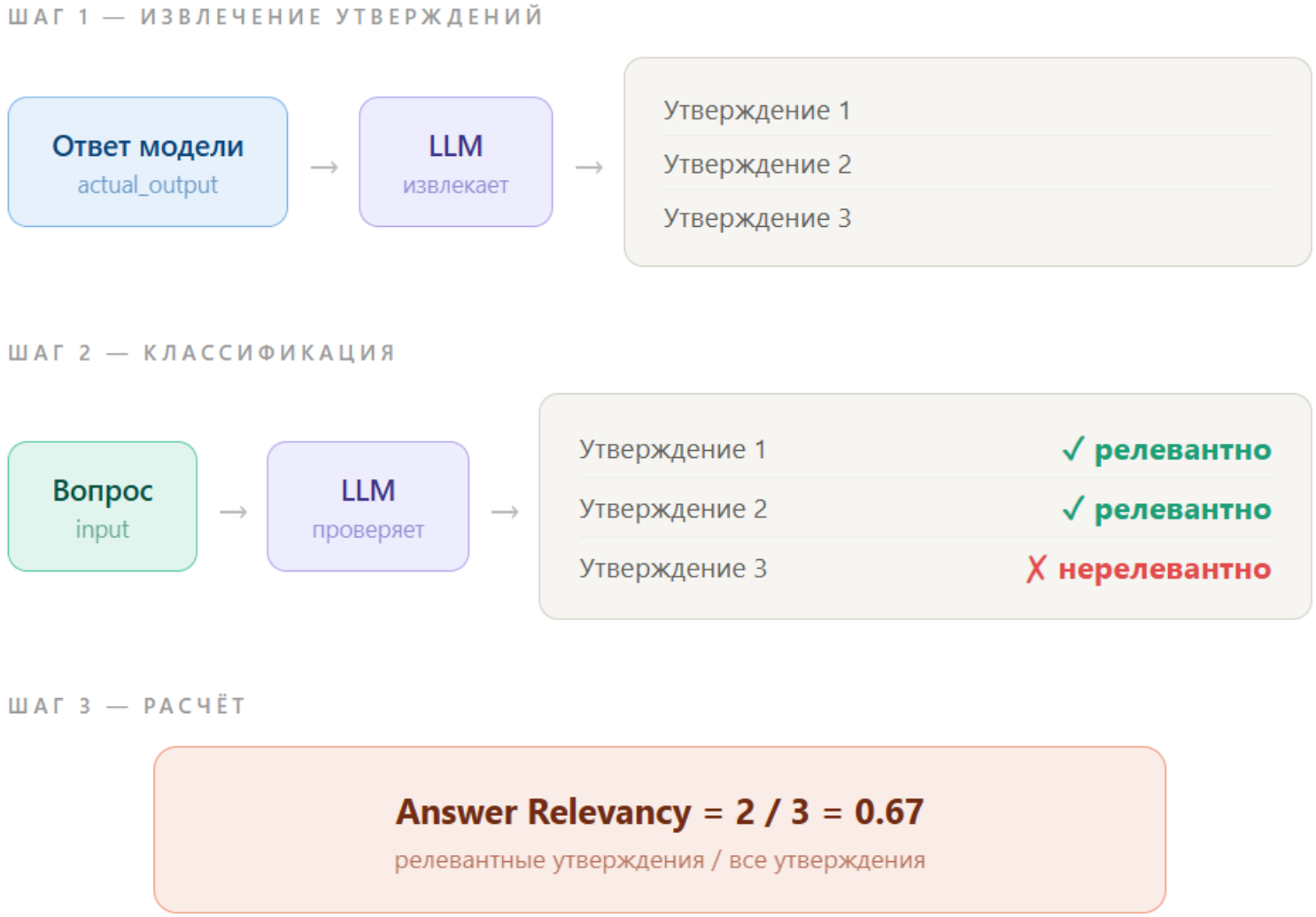
© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Answer Relevancy (релевантность ответа)

Оценка того, как хорошо наша система генерирует ответ.



© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Context Recall (полнота контекста)

Оценка того, как хорошо наша система находит контекст.

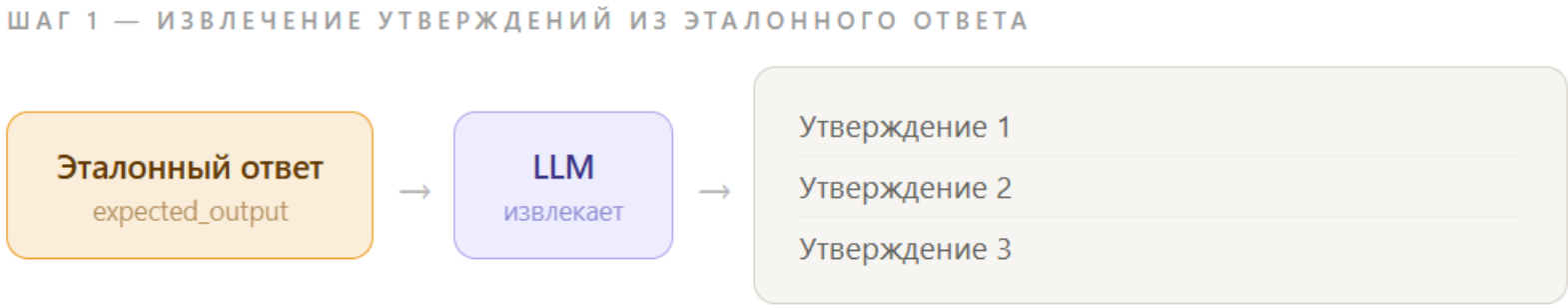
© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Contextual Recall (полнота контекста)

Оценка того, как хорошо наша система находит контекст.



© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Contextual Recall (полнота контекста)

Оценка того, как хорошо наша система находит контекст.

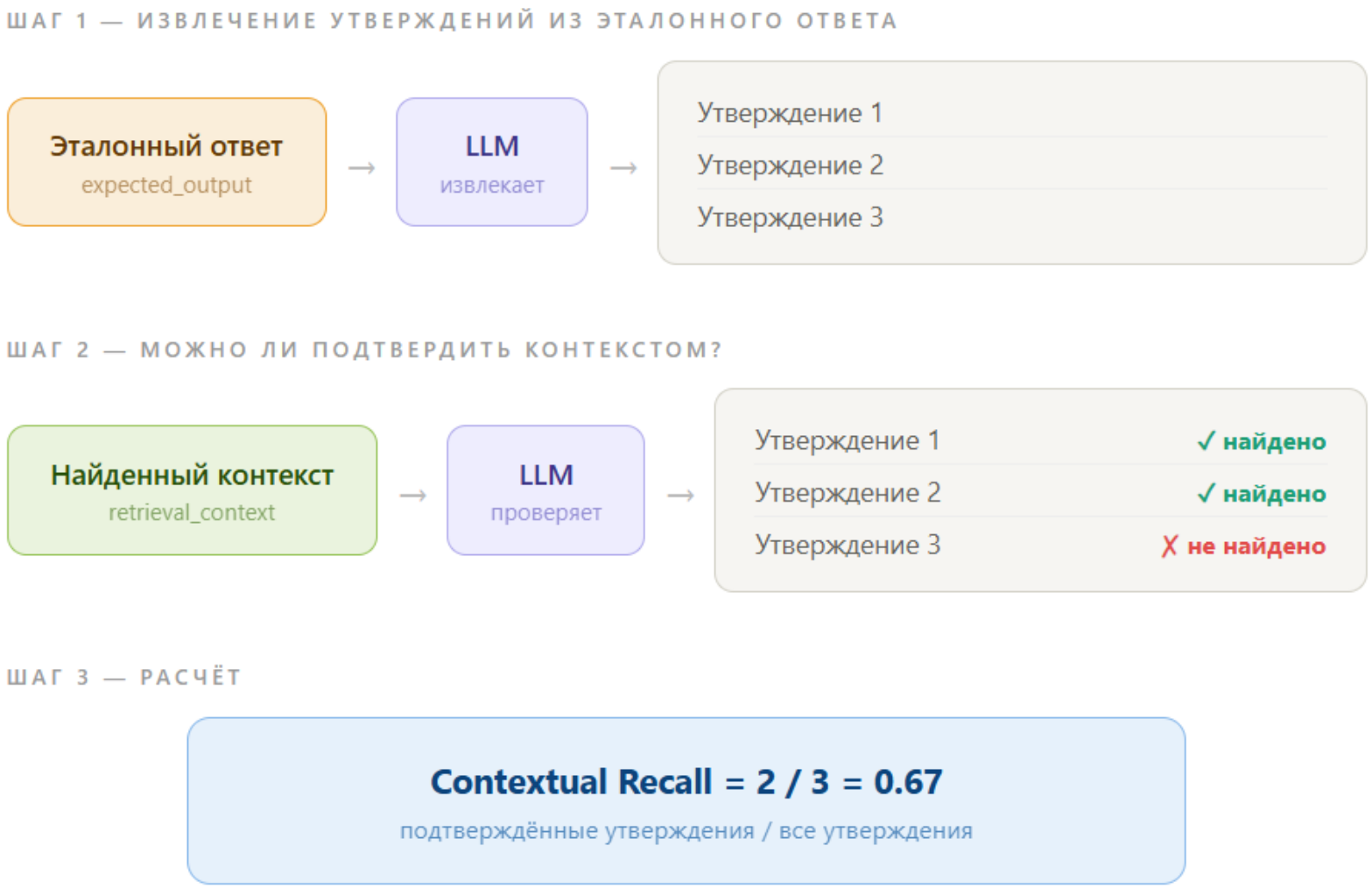
© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Выберем метрики для оценки RAG

Contextual Recall (полнота контекста)

Оценка того, как хорошо наша система находит контекст.



© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Метрика G-Eval = LLM-as-a-Judge + Chain-of-Thought

© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Метрика G-Eval = LLM-as-a-Judge + Chain-of-Thought

Описание задачи

Вам дан ответ RAG-ассистента супермаркета на запрос покупателя о товарах.

Оцените ответ по одной метрике.

Критерии оценки

Фактологическая точность (1–5) — соответствие ответа базе товаров:

— название и бренд
— цена и объём
— наличие на складе
— отсутствие выдуманных позиций

Шаги оценки

1. Прочитать запрос и контекст из базы.

2. Сверить ответ ассистента с контекстом.

3. Поставить оценку от 1 до 5: 1 — ответ противоречит базе, 5 — всё подтверждено.

© red_mad_robot,

https://deepeval.com/docs/metrics-llm-evals

Метрика G-Eval = LLM-as-a-Judge + Chain-of-Thought

Описание задачи
...

Критерии оценки
...

Шаги оценки
...

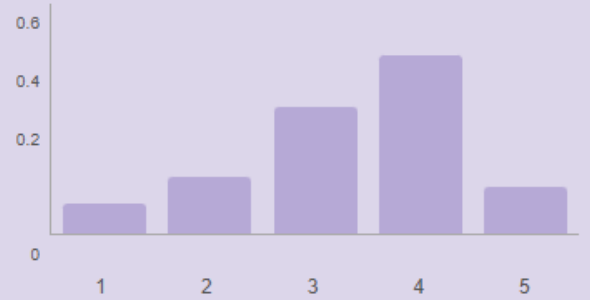
Входной контекст

Запрос: «Посоветуй безлактозное молоко до 150 ₽, которое есть в магазине на Ленина, 5». Фрагменты из базы: Молоко «Parmalat» безлактозное 1 л — 139 ₽, в наличии 12 шт.; Молоко «Валио» безлактозное 1 л — 189 ₽, в наличии 4 шт. ...

Целевой результат

Ответ ассистента: «В магазине на Ленина, 5 под ваш бюджет подходит молоко „Parmalat“ безлактозное 1 л за 139 ₽, сейчас в наличии 12 упаковок. „Валио“ стоит 189 ₽ и выходит за бюджет ...»

G-Eval



Взвешенная сумма: 3.41

https://deepeval.com/docs/metrics-llm-evals

© red_mad_robot,

Метрики в DeepEval

Инициализация LLM Judge

```python
import os
from deepeval.models.base_model import DeepEvalBaseLLM
from openai import OpenAI
from dotenv import load_dotenv

load_dotenv()

class CustomOpenAIModel(DeepEvalBaseLLM):
    def __init__(self, model_name: str, base_url: str, api_key: str):
        ...
    def load_model(self):
        ...
    def generate(self, prompt: str) -> str:
        ...
    async def a_generate(self, prompt: str) -> str:
        ...
    def get_model_name(self) -> str:
        ...

# Инициализируем judge-модель
judge_model = CustomOpenAIModel(
    model_name=os.getenv("OPENAI_MODEL"),
    base_url=os.getenv("OPENAI_BASE_URL"),
    api_key=os.getenv("OPENAI_API_KEY"),
)
```

© red_mad_robot,

Метрики в DeepEval

Инициализация LLM Judge

```python
import os
from deepeval.models.base_model import DeepEvalBaseLLM
from openai import OpenAI
from dotenv import load_dotenv

load_dotenv()

class CustomOpenAIModel(DeepEvalBaseLLM):
    def __init__(self, model_name: str, base_url: str, api_key: str):
        ...
    def load_model(self):
        ...
    def generate(self, prompt: str) -> str:
        ...
    async def a_generate(self, prompt: str) -> str:
        ...
    def get_model_name(self) -> str:
        ...

# Инициализируем judge-модель
judge_model = CustomOpenAIModel(
    model_name=os.getenv("OPENAI_MODEL"),
    base_url=os.getenv("OPENAI_BASE_URL"),
    api_key=os.getenv("OPENAI_API_KEY"),
)
```

© red_mad_robot,

Метрики в DeepEval

```python
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric
from custom_model import judge_model

# --- Метрика 1: Answer Relevancy ---
answer_relevancy = AnswerRelevancyMetric(
    threshold=0.5,
    model=judge_model,
)

# --- Метрика 2: Contextual Recall ---
contextual_recall = ContextualRecallMetric(
    threshold=0.5,
    model=judge_model,
)
```

© red_mad_robot,

Реализация G-Eval с DeepEval

```python
from deepeval.metrics import GEval

from custom_model import judge_model

# --- Метрика 3: GEval ---
correctness = GEval(
    name="Correctness",
    criteria=(
        "Оцени, насколько actual_output корректно и полно отвечает на вопрос "
        "по сравнению с expected_output. Учитывай фактическую точность: "
        "цены, названия, характеристики товаров должны совпадать."
    ),
    evaluation_params=[
        LLMTestCaseParams.INPUT,
        LLMTestCaseParams.ACTUAL_OUTPUT,
        LLMTestCaseParams.EXPECTED_OUTPUT,
    ],
    threshold=0.5,
    model=judge_model,
)
```

© red_mad_robot,

Реализация G-Eval с DeepEval

```python
from deepeval.metrics import GEval

from custom_model import judge_model

# --- Метрика 3: GEval ---
correctness = GEval(
    name="Correctness",
    criteria=(
        "Оцени, насколько actual_output корректно и полно отвечает на вопрос "
        "по сравнению с expected_output. Учитывай фактическую точность: "
        "цены, названия, характеристики товаров должны совпадать."
    ),
    evaluation_params=[
        LLMTestCaseParams.INPUT,
        LLMTestCaseParams.ACTUAL_OUTPUT,
        LLMTestCaseParams.EXPECTED_OUTPUT,
    ],
    threshold=0.5,
    model=judge_model,
)
```

© red_mad_robot,

Собираем всё вместе

© red_mad_robot

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

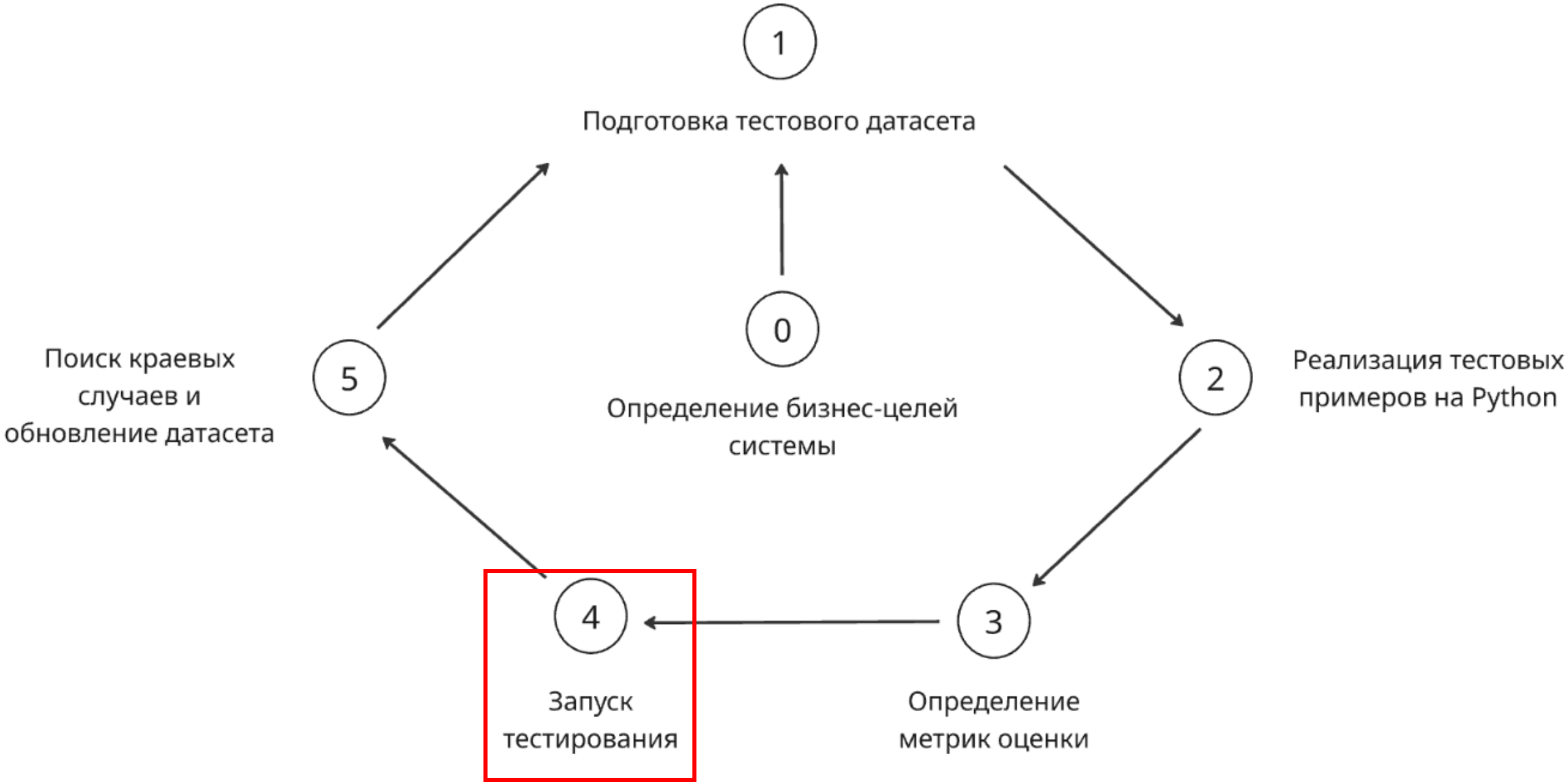
© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

© red_mad_robot,

Процесс тестирования с DeepEval



© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

```python
from deepeval import evaluate
from deepeval.dataset import EvaluationDataset, Golden
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric, GEval
from deepeval.test_case import LLMTestCase, LLMTestCaseParams

# Создание датасета из Golden-примеров
dataset = EvaluationDataset(goldens=[...])

# Инициализация метрик
answer_relevancy = AnswerRelevancyMetric(...)
contextual_recall = ContextualRecallMetric(...)
correctness = GEval(...)

# Тестовые примеры
test_cases = []
for golden in dataset.goldens:
    res, chunks = rag_pipeline(golden.input)
    test_cases.append(LLMTestCase(...))

# Запуск оценки
evaluate(
    test_cases=test_cases,
    metrics=[answer_relevancy, contextual_recall, correctness]
)
```

© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

```python
from deepeval import evaluate
from deepeval.dataset import EvaluationDataset, Golden
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric, GEval
from deepeval.test_case import LLMTestCase, LLMTestCaseParams

# Создание датасета из Golden-примеров
dataset = EvaluationDataset(goldens=[...])

# Инициализация метрик
answer_relevancy = AnswerRelevancyMetric(...)
contextual_recall = ContextualRecallMetric(...)
correctness = GEval(...)

# Тестовые примеры
test_cases = []
for golden in dataset.goldens:
    res, chunks = rag_pipeline(golden.input)
    test_cases.append(LLMTestCase(...))

# Запуск оценки
evaluate(
    test_cases=test_cases,
    metrics=[answer_relevancy, contextual_recall, correctness]
)
```

© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

```python
from deepeval import evaluate
from deepeval.dataset import EvaluationDataset, Golden
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric, GEval
from deepeval.test_case import LLMTestCase, LLMTestCaseParams

# Создание датасета из Golden-примеров
dataset = EvaluationDataset(goldens=[...])

# Инициализация метрик
answer_relevancy = AnswerRelevancyMetric(...)
contextual_recall = ContextualRecallMetric(...)
correctness = GEval(...)

# Тестовые примеры
test_cases = []
for golden in dataset.goldens:
    res, chunks = rag_pipeline(golden.input)
    test_cases.append(LLMTestCase(...))

# Запуск оценки
evaluate(
    test_cases=test_cases,
    metrics=[answer_relevancy, contextual_recall, correctness]
)
```

© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

```python
from deepeval import evaluate
from deepeval.dataset import EvaluationDataset, Golden
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric, GEval
from deepeval.test_case import LLMTestCase, LLMTestCaseParams

# Создание датасета из Golden-примеров
dataset = EvaluationDataset(goldens=[...])

# Инициализация метрик
answer_relevancy = AnswerRelevancyMetric(...)
contextual_recall = ContextualRecallMetric(...)
correctness = GEval(...)

# Тестовые примеры
test_cases = []
for golden in dataset.goldens:
    res, chunks = rag_pipeline(golden.input)
    test_cases.append(LLMTestCase(...))

# Запуск оценки
evaluate(
    test_cases=test_cases,
    metrics=[answer_relevancy, contextual_recall, correctness]
)
```

© red_mad_robot,

Pipeline тестирования с DeepEval

Собираем все вместе – получаем pipeline тестирования

```python
from deepeval import evaluate
from deepeval.dataset import EvaluationDataset, Golden
from deepeval.metrics import AnswerRelevancyMetric, ContextualRecallMetric, GEval
from deepeval.test_case import LLMTestCase, LLMTestCaseParams

# Создание датасета из Golden-примеров
dataset = EvaluationDataset(goldens=[...])

# Инициализация метрик
answer_relevancy = AnswerRelevancyMetric(...)
contextual_recall = ContextualRecallMetric(...)
correctness = GEval(...)

# Тестовые примеры
test_cases = []
for golden in dataset.goldens:
    res, chunks = rag_pipeline(golden.input)
    test_cases.append(LLMTestCase(...))

# Запуск оценки
evaluate(
    test_cases=test_cases,
    metrics=[answer_relevancy, contextual_recall, correctness]
)
```

© red_mad_robot,

Pipeline тестирования с DeepEval

Запуск

```
python .\run_evaluation.py
```

© red_mad_robot,

Pipeline тестирования с DeepEval

```
======================================================================

Overall Metric Pass Rates

Answer Relevancy: 60.00% pass rate
Contextual Recall: 100.00% pass rate
Correctness [GEval]: 100.00% pass rate

======================================================================
```

© red_mad_robot,

Pipeline тестирования с DeepEval

```
==========================================================================

Metrics Summary

  - ❌ Answer Relevancy (score: 0.14285714285714285, threshold: 0.5, strict: False, evaluation model: google/gemini-2.5-flash-lite-preview-09-2025, reason: The score is 0.14
because the entire output consisted only of greetings, general pleasantries, feature descriptions (sound quality, battery life, transparency mode), and closing remarks, compl
etely fa
  - ✅ C
ire expe
 руб.' f
  - ✅ C
ers the
s detail
mpletene

For test

  - inpu
  - actu

Наушники

Metrics Summary

  - ✅ Answer Relevancy (score: 0.8, threshold: 0.5, strict: False, evaluation model: google/gemini-2.5-flash-lite-preview-09-2025, reason: The score is 0.80 because whi
e core question about temperature-maintaining kettles was addressed, the presence of unnecessary greetings detracted slightly from the direct relevance of the overall ou
, error: None)
  - ✅ Contextual Recall (score: 1.0, threshold: 0.5, strict: False, evaluation model: google/gemini-2.5-flash-lite-preview-09-2025, reason: The score is 1.00 because th
ire expected output sentence, stating that the SteelKettle 1.7L electric kettle maintains a temperature between 70–95°C, is directly verifiable via the information found
ode 1 of the node(s) in retrieval context., error: None)
  - ✅ Correctness [GEval] (score: 0.6, threshold: 0.5, strict: False, evaluation model: google/gemini-2.5-flash-lite-preview-09-2025, reason: The Actual Output correctl
ntifies the product that supports temperature ('Электрический чайник «SteelKettle 1.7L»') and includes the temperature range (70°C до 95°C), aligning with the factual de
 in the Expected Output. However, the Actual Output is significantly more verbose and introduces an unrelated product (a thermos), failing to meet the concise standard s
 the Expected Output, which only required the single sentence answer., error: None)

For test case:

  - input: Какой чайник поддерживает температуру?
  - actual output: Здравствуйте! Рад помочь вам с выбором.

Насколько я понимаю, вы ищете прибор, который может поддерживать определённую температуру напитка.
```

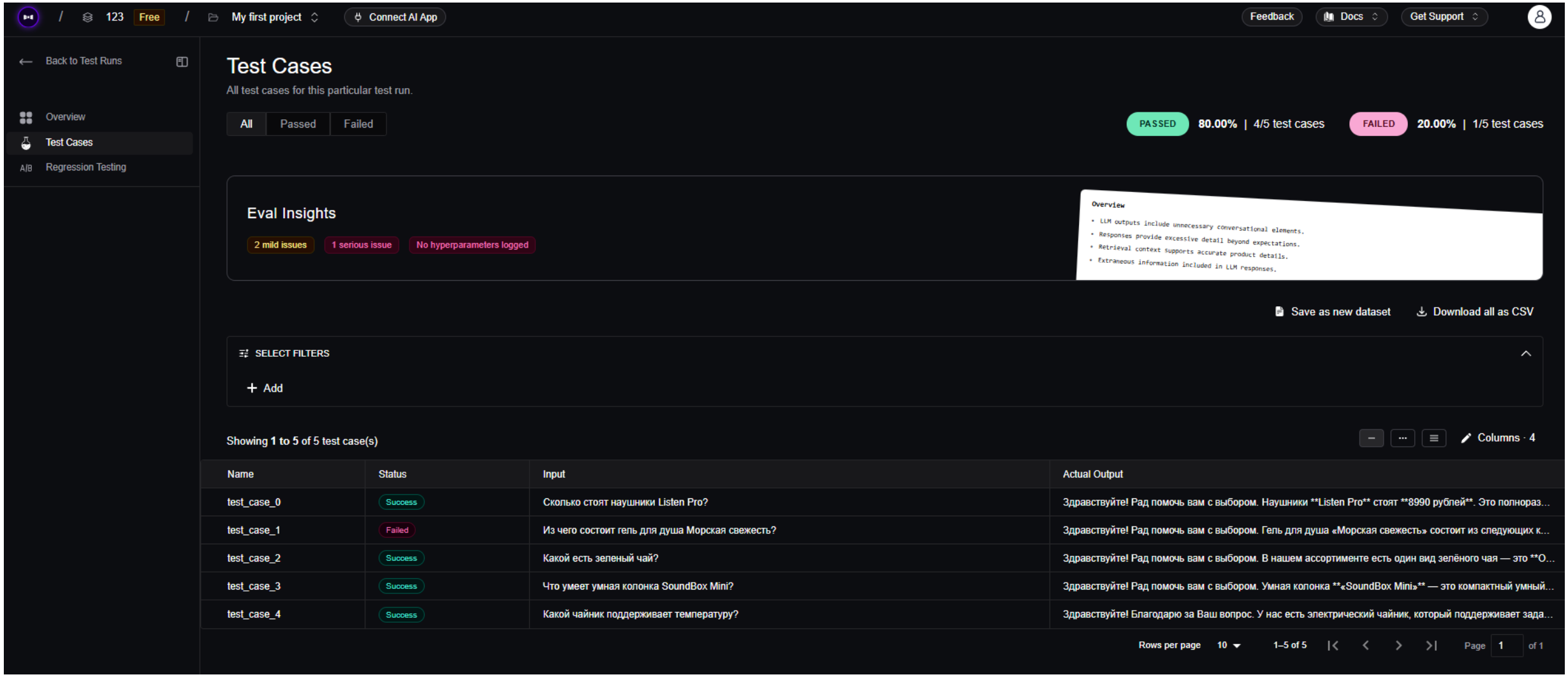
© red_mad_robot,

Процесс тестирования с DeepEval

© red_mad_robot,

Confident-AI

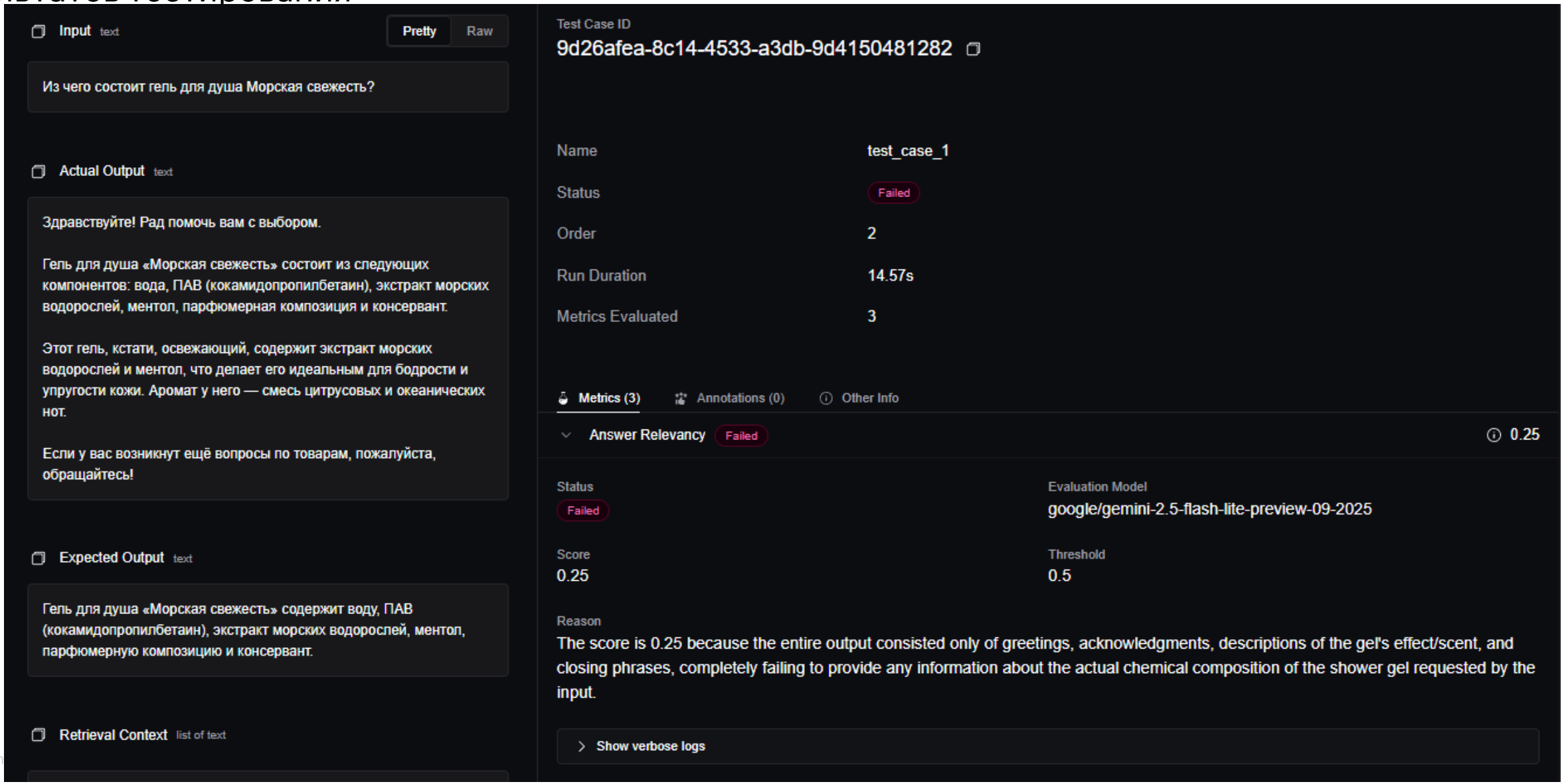
Confident-AI – графический интерфейс для отслеживания результатов тестирования



© red_mad_robot,

Confident-AI

Confident-AI – графический интерфейс для отслеживания результатов тестирования



© red_m

Confident-AI

Confident-AI – графический интерфейс для отслеживания результатов тестирования

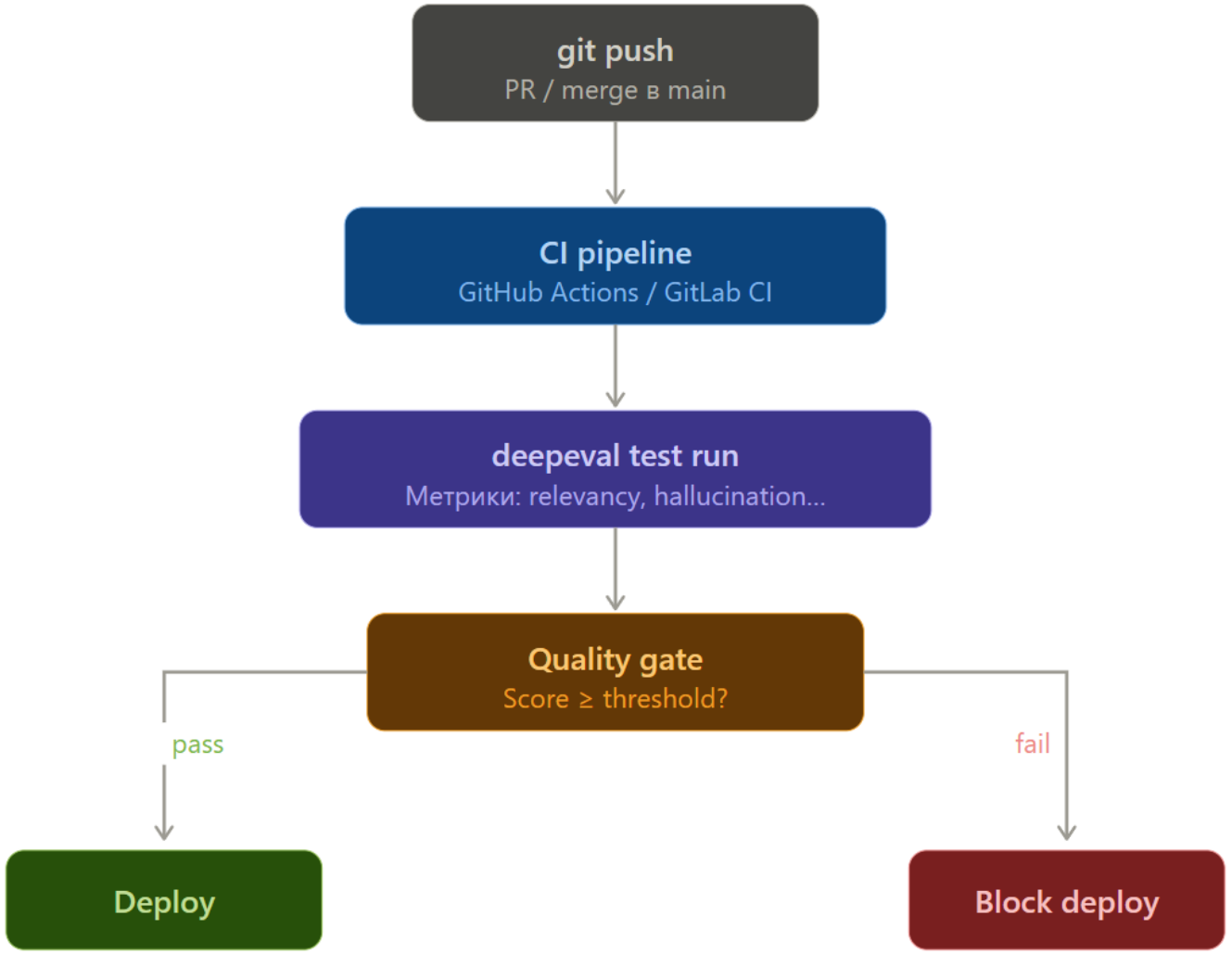
Один из основных минусов – **только облачное решение**. Для конфиденциальных данных не подойдет.

© red_mad_robot,

Использование DeepEval в CI/CD

© red_mad_robot

Внедрение DeepEval в CI\CD



© red_mad_robot,

Внедрение DeepEval в CI\CD

```yaml
name: RAG Evaluation

on:
  push:
    branches: [main]
  pull_request:
    branches: [main]
  workflow_dispatch:

env:
  PYTHON_VERSION: "3.12"
  EMBEDDING_MODEL: "deepvk/USER-base"
```

© red_mad_robot,

Внедрение DeepEval в CI\CD

```yaml
jobs:
  evaluate:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v4

      - name: Set up Python
        uses: actions/setup-python@v5
        with:
          python-version: ${{ env.PYTHON_VERSION }}

      - name: Cache pip packages
        uses: actions/cache@v4
        with:
          path: ~/.cache/pip
          key: pip-${{ runner.os }}-${{ hashFiles('requirements.txt') }}
          restore-keys: pip-${{ runner.os }}-
```

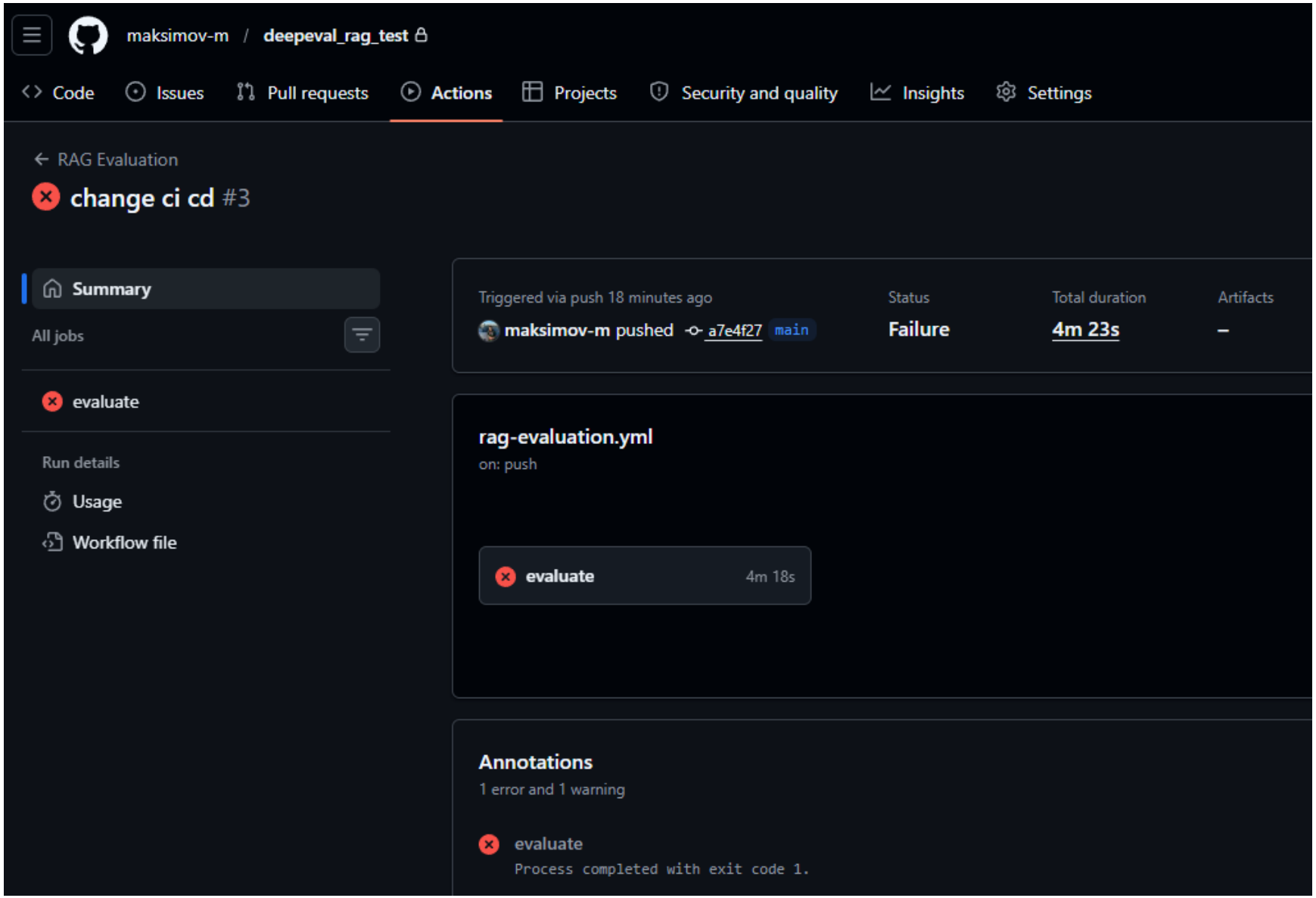
© red_mad_robot,

Внедрение DeepEval в CI\CD

```
      - name: Cache embedding model
        uses: actions/cache@v4
        with:
          path: ~/.cache/huggingface
          key: hf-${{ env.EMBEDDING_MODEL }}
          restore-keys: hf-

      - name: Download embedding model
        run: python -c "from sentence_transformers import SentenceTransformer; SentenceTransformer('${{
env.EMBEDDING_MODEL }}')"

      - name: Run DeepEval tests
        env:
          OPENAI_API_KEY: ${{ secrets.OPENAI_API_KEY }}
          OPENAI_BASE_URL: ${{ secrets.OPENAI_BASE_URL }}
          OPENAI_DEFAULT_MODEL: ${{ secrets.OPENAI_DEFAULT_MODEL }}
        run: deepeval test run test_rag.py
```

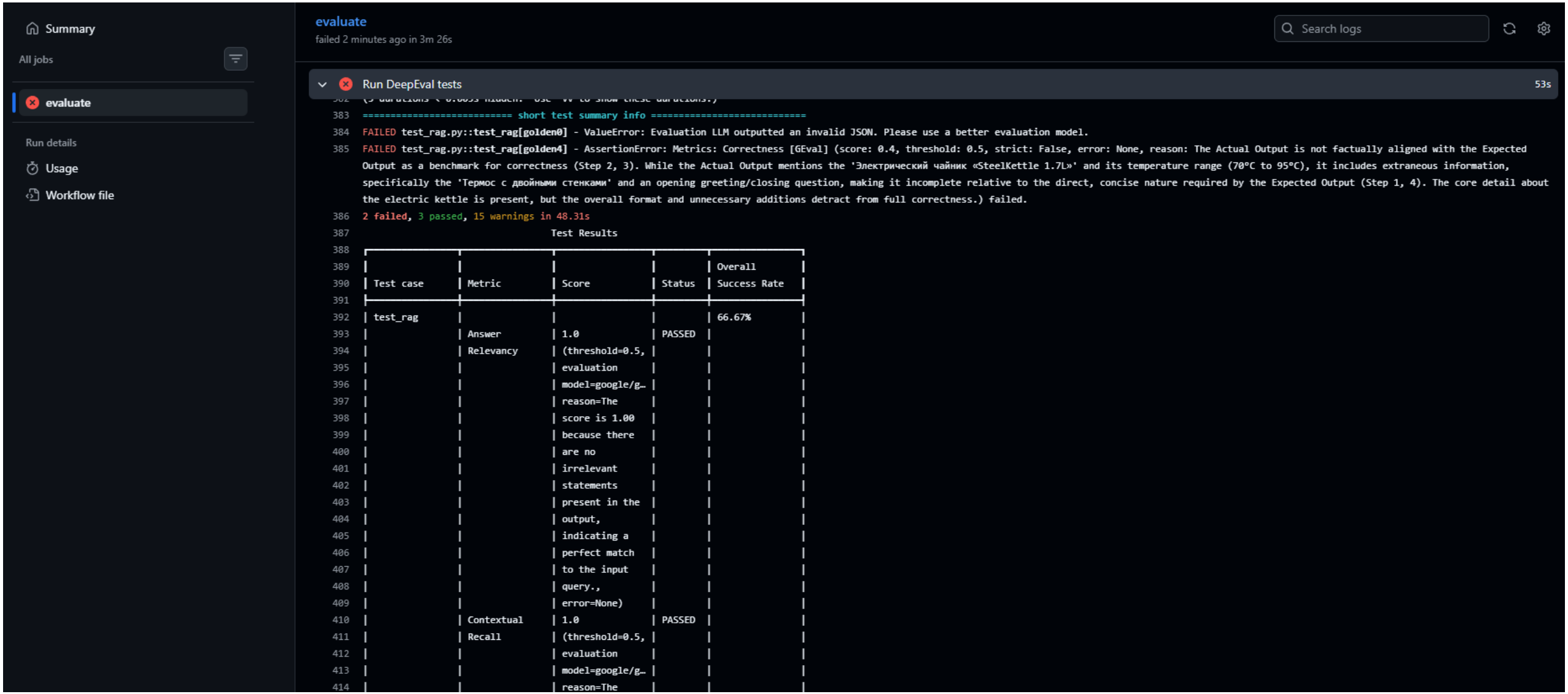
© red_mad_robot,

Внедрение DeepEval в CI\CD



© red_mad_robot,

Внедрение DeepEval в CI\CD



© red_mad_robot,

Внедрение DeepEval в CI\CD

825		Correctness	0.4	FAILED		
826		[GEval]	(threshold=0.5,			
827			evaluation			
828			model=google/g…			
829			reason=The			
830			Actual Output			
831			is not			
832			factually			
833			aligned with			
834			the Expected			
835			Output as a			
836			benchmark for			
837			correctness			
838			(Step 2, 3).			
839			While the			

© red_mad_robot,

Заключение

1. Большая, дорогая модель в качестве судьи - не значит лучшая.
2. Работа с LLM-приложениями требует от тестировщиков новых навыков, которые необходимо осваивать уже сейчас.
3. Подбирайте метрики в соответствии с «узким» местом своей системы.

© red_mad_robot,

Спасибо за внимание!

Здесь много интересного!

© red_mad_robot