MWS AI AGENTS PLATFORM

Инструкция по эксплуатации после установки

Версия 2.1

22 июня 2026 г.

СОДЕРЖАНИЕ

ОСНОВНЫЕ ПОНЯТИЯ И ТЕРМИНЫ

Агент

Автономная система, использующая LLM для сбора данных, анализа, принятия решений и выполнения задач различной сложности.

База знаний RAG

Набор документов, загруженных в систему, используемый RAG для выдачи релевантных ответов пользователю.

Блок активации (активационный блок)

Блок, содержащий условие активации сценария, к которому он относится. Условие активации проверяется относительно запроса пользователя.

Блок активации event

Блок, который запускает сценарий при наступлении определённого события, такого как начало диалога (init), пользовательское событие на поверхности, подключенной через канал HTTP, или отсутствие совпадения (no_match).

Блок активации intent

Блок, запускающий сценарий, если в сообщении пользователя распознан определённый интент.

Блок активации match

Блок, запускающий сценарий, если сообщение пользователя соответствует заданному регулярному выражению.

Блок реакции (реакционный блок)

Блок, содержащий логику, которая выполняется при активации сценария. Если сценарий содержит несколько блоков реакции, они выполняются последовательно.

Бот

Программа для имитации общения с пользователями через голосовые или текстовые интерфейсы, предназначенная для предоставления информации или выполнения задач.

Веб-клиент AI Agents Platform

Пользовательский веб-интерфейс для работы с платформой через браузер. Позволяет создавать и адаптировать ботов под конкретные требования.

Датасет

Набор данных для обучения, тестирования и оценки ML-моделей.

Движок (agent-engine)

Сервис для обработки запросов пользователей с помощью ботов, созданных в no-code-конструкторе.

Индекс RAG

Хранилище фактологической информации для использования в RAG, содержащее векторную часть (embedding) и текстовое представление, позволяющее проводить быстрый поиск релевантных записей.

Интент

Намерение пользователя, отражённое в поисковом запросе.

Краулинг (crawling)

Процесс автоматического сканирования, обхода и скачивания веб-страниц с помощью специальных сервисов — краулеров. Краулинг используется для сбора информации с веб-сайтов, анализа и индексации контента.

Классификатор (Classifier)

Модель машинного обучения, которая автоматически обучается и оптимизируется для решения задач классификации.

Модель машинного обучения (ML-модель)

Алгоритмическая конструкция для прогнозирования или классификации данных на основе обучения.

Паттерн

Формальное правило, описывающее ключевые слова и выражения для классификации запросов пользователя.

Проект

Совокупность сценариев разработанного бота или агента. Может содержать несколько версий.

Промпт

Подсказка для нейросети о том, что именно от неё требуется.

Рабочий процесс Langflow

Процесс в рамках фреймворка Langflow, интегрирующий несколько компонентов для обработки и генерации текста на основе пользовательских запросов и технологии RAG.

Сценарий

Логика работы проекта, направленная на достижение определённой цели в диалоге с клиентом. В no-code-конструкторе сценарий представляет собой последовательность двух и более связанных между собой нод.

Трейс

В трейсинге — полная последовательность событий для одного запроса (например, от клиента к серверу и обратно через несколько микросервисов)

Трейсинг

Метод мониторинга и отладки в распределённых системах, который позволяет отслеживать полный путь запроса через несколько сервисов, компонентов или узлов.

Узел (нода, node)

Компонент сценария, который объединяет активационные или реакционные блоки для их последовательного выполнения: может включать в себя один или несколько блоков.

AI-сервис

Сервисы, использующие искусственный интеллект для обработки информации и решения задач в различных сферах.

AncSetFit (Anchored SetFit)

Расширение способа обучения ML-моделей SetFit, использующее «якорные» (anchor) примеры для каждого класса, что обеспечивает более стабильное обучение и лучшую генерализацию при экстремально малом количестве данных.

AutoML

Автоматизированный процесс создания и улучшения моделей машинного обучения.

Autoscaling (автомасштабирование)

Автоматическое изменение количества работающих экземпляров (инстансов) модели в зависимости от текущей нагрузки и потребления ресурсов.

Batching (батчинг)

Подход, при котором несколько отдельных запросов объединяются в одну группу (батч), чтобы выполнить их одновременно за одно обращение к модели.

Embedding (Эмбеддинг, вектор)

Векторное представление слова или фразы, полученное из моделей обработки естественного языка.

Fine-tuning

Дообучение предварительно обученной модели на новых данных путём обновления всех или части весов модели и/или добавления новых слоёв для адаптации к специфической задаче.

Keycloak

Стороннее решение для идентификации пользователей и контроля доступа.

Langflow

Фреймворк для создания агентов с помощью no-code конструктора.

Large language model (LLM)

Языковая модель, обученная на больших объёмах текста для выполнения задач NLP.

Named Entity Recognition (NER)

Модель машинного обучения для выделения и классификации именованных сущностей в тексте.

Natural Language Processing (NLP)

Область искусственного интеллекта, занимающаяся обработкой и анализом естественного языка.

Retrieval Augmented Generation (RAG)

Технология ML, объединяющая поиск информации и генерацию текста для создания ответов на запросы.

SetFit

Метод эффективного обучения ML-моделей на малом количестве данных, использующий обучение через сравнение примеров из разных классов и последующее обучение классификатора на полученных векторных представлениях (эмбеддингах) без дообучения всей модели.

Slug (слаг)

Уникальный ключ, который представляет собой имя сценария строчными буквами. Вместо пробелов в качестве разделителей используются знаки нижнего подчеркивания.

Webim

Омниканальная платформа для коммуникаций с клиентами. Подробнее см. https://webim.ru.

О платформе

AI Agents Platform — это платформа для создания AI решений, которые автоматизируют обслуживание клиентов через коммуникационные каналы и поддерживают омниканальное взаимодействие. Особенность AI Agents Platform — набор готовых инструментов, не требующих от пользователей опыта в машинном обучении или навыков программирования. Благодаря этому клиенты платформы могут

самостоятельно управлять жизненным циклом ботов и AI-агентов, создавать и обслуживать ML-модели и AI-сервисы. В зависимости от варианта поставки платформа AI Agents Platform предоставляет следующие возможности:

- создание сценария бота или агента в удобном no-code-конструкторе;
- обработка логики бота на высокопроизводительном и отказоустойчивом движке;
- подключение ботов к каналам взаимодействия с поверхностями;
- подключение и использование LLM к ботам; создание и обучение ML-моделей типов Классификатор и NER;
- разметка данных для формирования и совершенствования обучающих датасетов;
- создание RAG-сервисов для быстрого поиска ответов на вопросы в регулярно обновляемой базе знаний.

Начало и завершение работы

Вход в веб-клиент

При первом входе в веб-клиент запросите имя пользователя и пароль, а также адрес для входа.

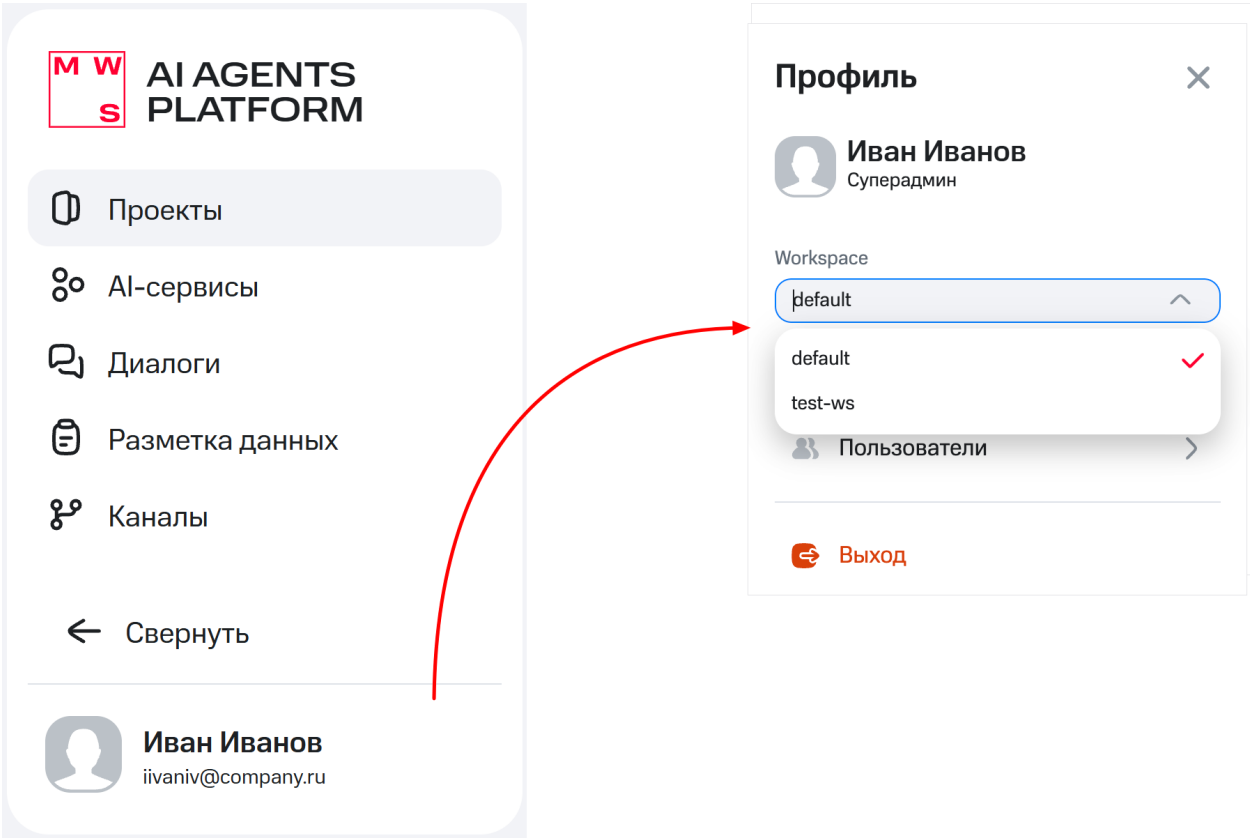
1. В браузере перейдите по полученному URL-адресу.
2. В окне авторизации введите данные вашей учетной записи.
3. В результате открывается стартовая страница.

Выбор воркспейса для работы

В платформе может быть несколько воркспейсов для работы. Воркспейс — это логическая область, предназначенная для разработки, настройки и использования продуктов и их ресурсов. Содержит ресурсы платформы, связывает пользователей с ресурсами через назначение ролей. Воркспейсы логически изолированы друг от друга. По умолчанию создан воркспейс с именем default. При необходимости администратор может добавить другие через базу данных.

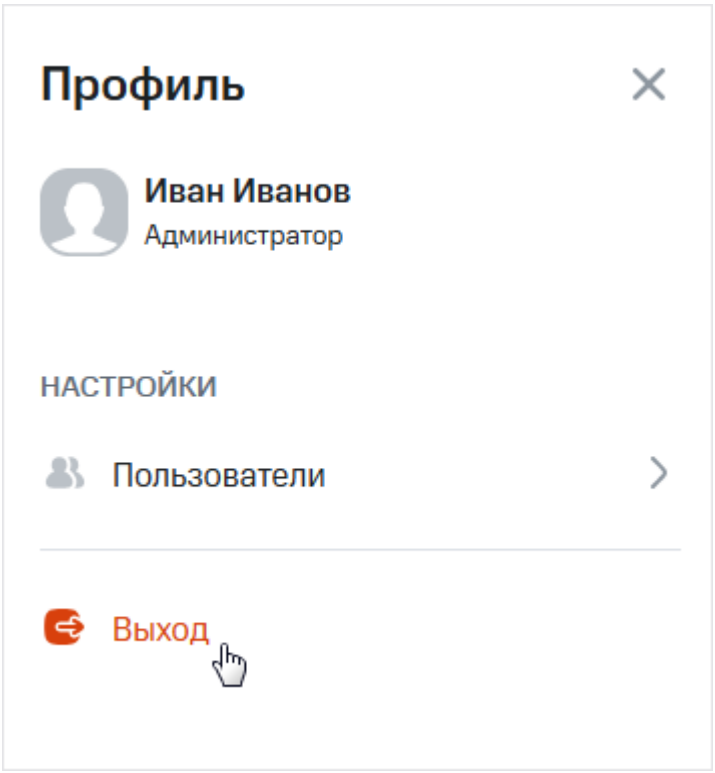
Чтобы выбрать воркспейс для работы:

1. Перейдите в профиль пользователя. Для этого нажмите на изображение профиля в левой нижней части системы.
2. Нажмите на иконку воркспейса. В выпадающем списке выберите нужный пункт:



Выход из системы

1. Нажмите на иконку профиля в нижнем левом углу веб-клиента.
2. В выпадающем списке выберите пункт **Выход**.
3. Нажмите на **Выйти**.



Настройки доступа

Для разграничения доступа в MWS AI Agents Platform предусмотрены воркспейсы — логическая область, предназначенная для разработки, настройки и использования продуктов и их ресурсов. Содержит ресурсы платформы, связывает пользователей с ресурсами через назначение ролей. Воркспейсы логически изолированы друг от друга. По умолчанию создан воркспейс с именем default. При необходимости администратор может добавить другие через базу данных.

После установки платформы первым в систему заходит пользователь с правами роли Суперадмин. Он имеет полный доступ ко всем ресурсам платформы, является сервисным аккаунтом. При первом входе суперадмин может добавить в роли других администраторов системы для дальнейшего управления доступом пользователей. Например, администратор бизнес-аккаунта или администратор воркспейса могут назначить роли пользователям в определенном бизнес-аккаунте или воркспейсе соответственно.

Для пользователей доступны роли:

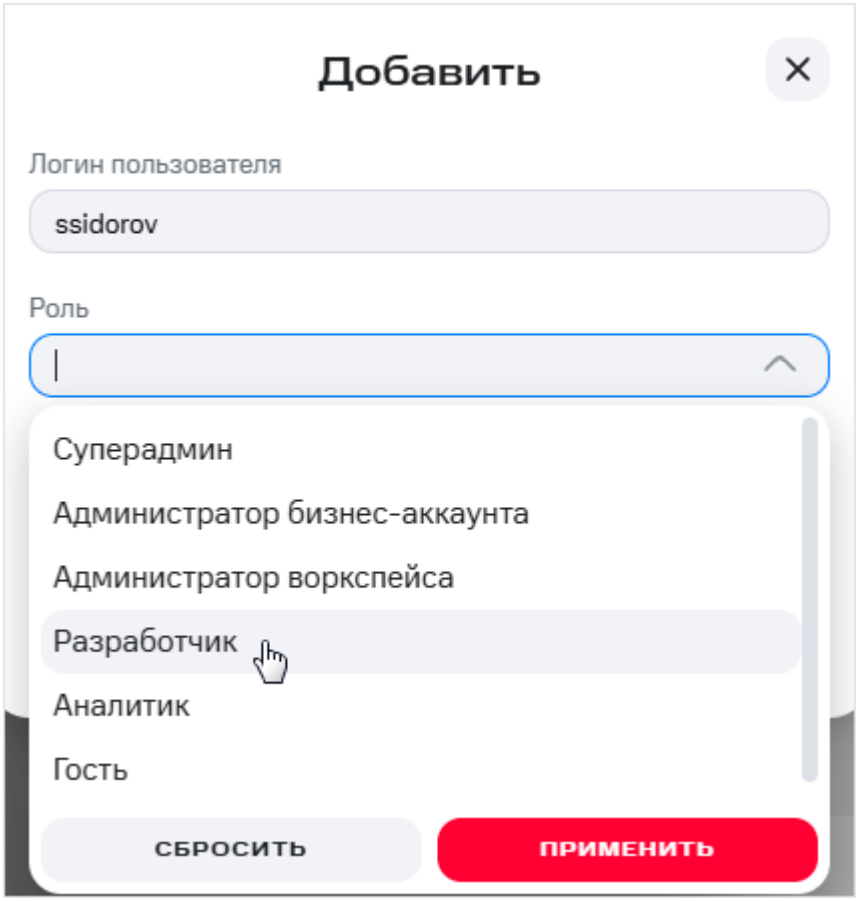
РОЛЬ	ОПИСАНИЕ	РЕЗРЕШЕНИЯ
Суперадмин	Имеет полный доступ ко всем ресурсам IAM, платформы и продуктов. Сервисный аккаунт	Полный доступ ко всем ресурсам платформы
Разработчик	Работа с объектами воркспейса	- получение информации о пользователе IAM; - получение списка пользователей IAM; - получение списка ролей IAM; - чтение своих собственных данных; - получение списка пользователей в воркспейсах IAM; - все операции с воркспейсами платформы
Аналитик	Чтение проектов и редактирование разметки	- получение информации о пользователе IAM; - чтение своих собственных данных; - чтение проектов воркспейсов платформы; - чтение диалогов воркспейсов платформы; - разметка диалогов воркспейсов платформы; - экспорт данных диалогов воркспейсов платформы; - чтение проектов разметки воркспейсов платформ; - чтение AI-сервисов воркспейсов платформы; - чтение каналов воркспейсов платформы

РОЛЬ	ОПИСАНИЕ	РЕЗРЕШЕНИЯ
Гость	Чтение диалогов проектов	- получение информации о пользователе IAM; - чтение своих собственных данных; - чтение проектов воркспейсов платформы; - чтение диалогов воркспейсов платформы
Администратор бизнес-аккаунта	Управление воркспейсами и их ресурсами на уровне бизнес-аккаунта	- получение информации о пользователе IAM; - получение списка пользователей IAM; - получение списка ролей IAM; - чтение своих собственных данных; - управление пользователями IAM, включая просмотр, назначение и отмену администратора; - все операции с воркспейсами IAM; - все операции с воркспейсами платформы
Администратор воркспейса	Управление всеми ресурсами воркспейса	- получение информации о пользователе IAM; - получение списка пользователей IAM; - получение списка ролей IAM; - чтение своих собственных данных; - все операции с пользователями воркспейсов IAM; - все операции с ролями воркспейсов IAM; - все операции с воркспейсами платформы

Добавление пользователя в роль

Чтобы включить пользователя в роль, у вас должны быть права администратора.

1. Перейдите в свой профиль. Для этого нажмите на картинку пользователя в нижнем левом углу веб-клиента.
2. Перейдите в список **Пользователи**.
3. Нажмите на кнопку **Добавить пользователя**.
4. В открывшемся окне введите логин пользователя и выберите роль, которую нужно назначить выбранному пользователю:

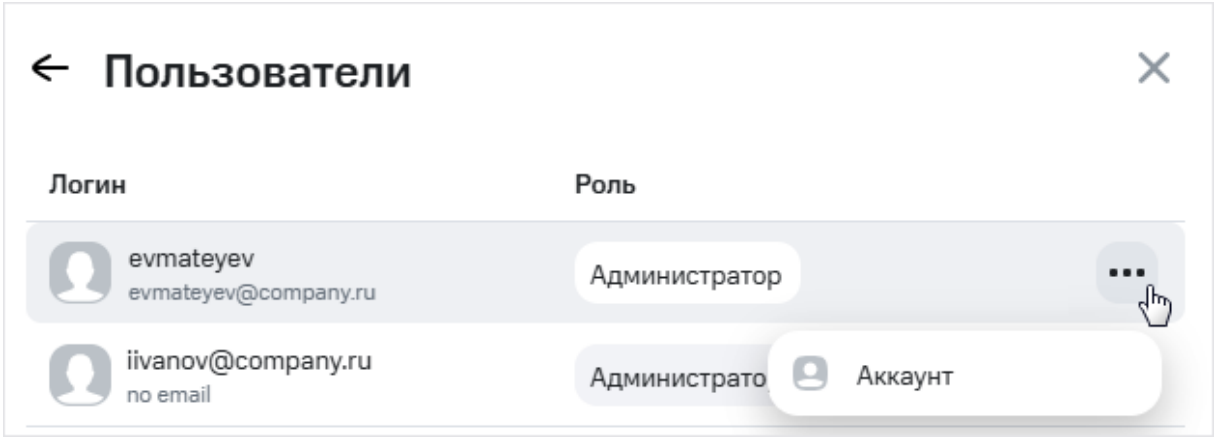


5. Нажмите на кнопку **Применить**.

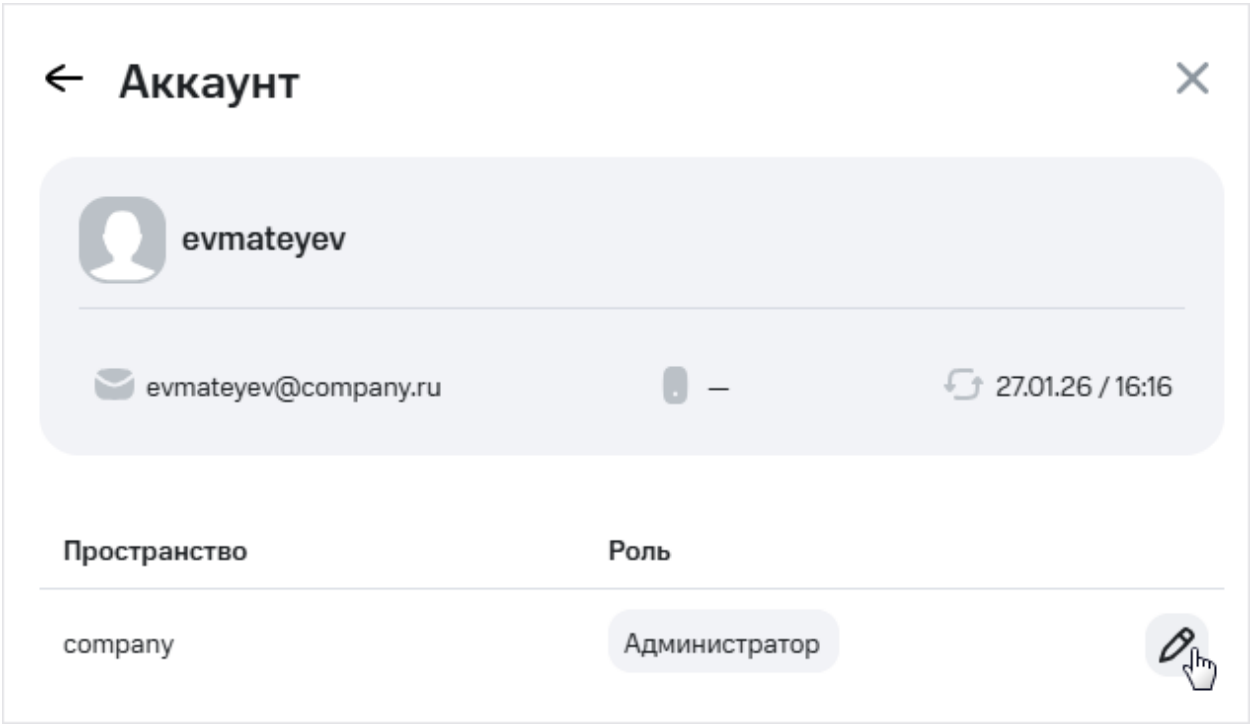
Изменение роли пользователя

Чтобы изменить роль пользователя:

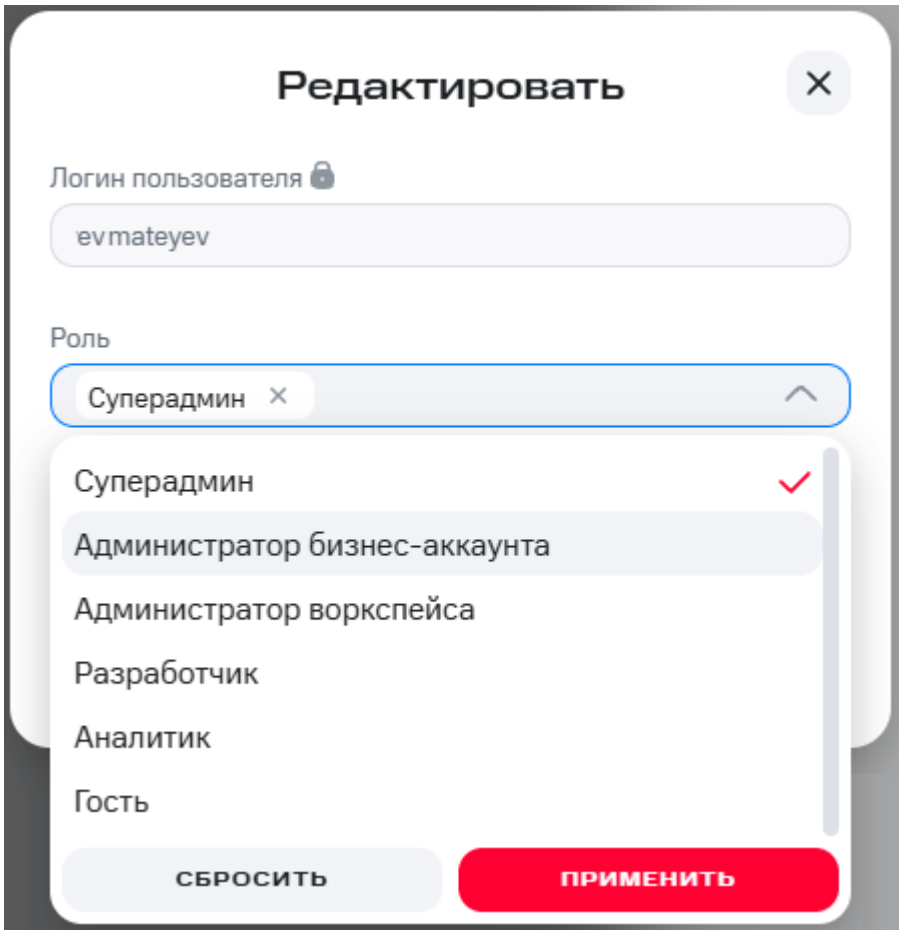
1. Перейдите в свой профиль. Для этого нажмите на картинку пользователя в нижнем левом углу веб-клиента.
2. Перейдите в список **Пользователи**.
3. Вызовите контекстное меню нужного пользователя и выберите пункт **Аккаунт**:



4. В открывшемся профиле нажмите на кнопку , чтобы скорректировать роль:



1. В выпадающем списке **Роль** выберите новую роль и нажмите на кнопку **Применить**. Один пользователь может быть включен сразу в несколько ролей. Чтобы удалить роль, нажмите на × рядом с названием.



1. Нажмите на кнопку **Сохранить**.

В результате в профиле пользователя отображаются выбранные роли.

Удаление пользователя

Чтобы удалить пользователя, у вас должны быть права администратора. Из своего профиля перейдите в список **Пользователи** и вызовите контекстное меню в строке нужного пользователя. Нажмите **Удалить**.

(пусто)

Работа с проектами

Для создания и редактирования проектов используется специальный конструктор сценариев. Он представляет собой визуальный редактор с предопределенным набором блоков. Достаточно перетащить их в рабочую область и заполнить поля и связи. Проект представляет собой совокупность сценариев разработанного бота или агента. Может содержать несколько версий.

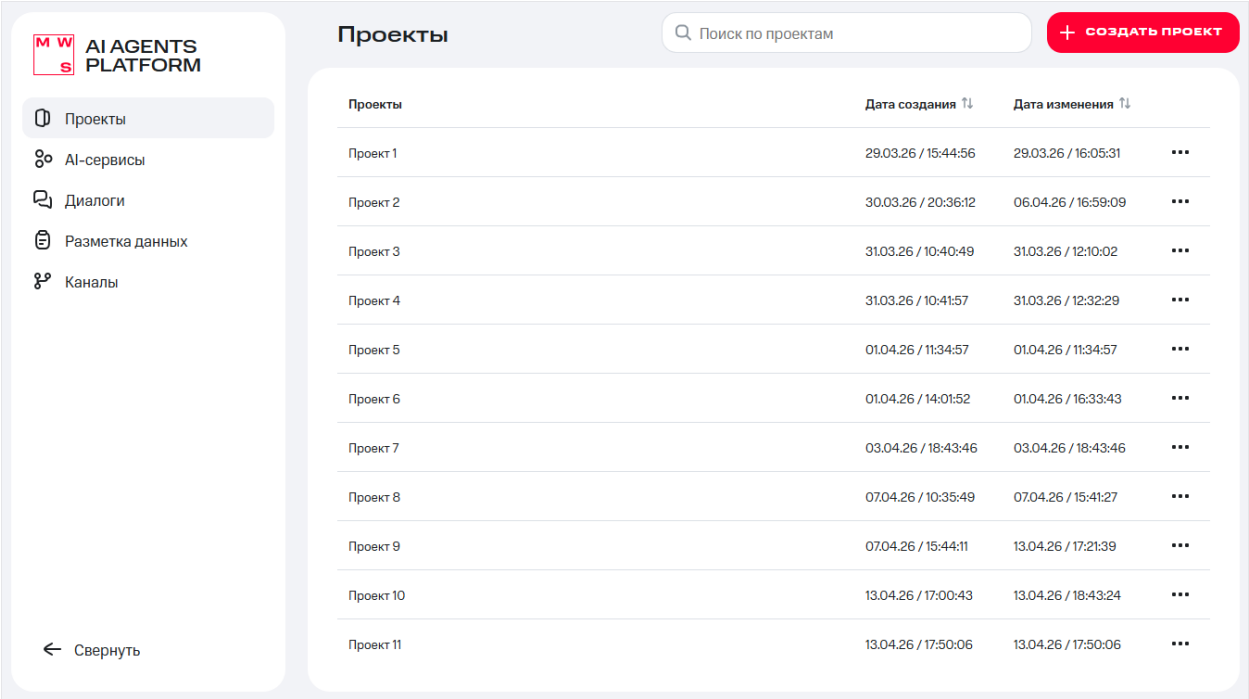
No-code-конструктор (конструктор сценариев) – это графический редактор в веб-клиенте AI Agents Platform, который позволяет создавать, просматривать и редактировать диалоговые сценарии бота или агента без разработки кода.

Бот – это программа, предназначенная для имитации общения с пользователями с помощью голосовых или текстовых интерфейсов. Его основная задача – предоставить информацию или выполнить задачи, которые в обычной жизни решаются через общение с человеком. Проект бота представляет собой совокупность заранее написанных сценариев, которые активируются в зависимости от определенного действия пользователя.

Созданные боты работают на едином движке. Загрузка, сохранение и исполнение логики выполняются по определенным правилам. После создания бота его настройки формируются автоматически и их можно сохранить в файл формата JSON.

Агент – это обработчик запроса для передачи ответа на естественном языке. Агент является оберткой над ML-моделью с механизмом Function calling, позволяющим вызывать внешние инструменты для выполнения различных задач. Если для обработки запроса пользователя нужно использовать ML-модели, то создайте проект агента и подключите к нему внешние инструменты.

Созданные проекты отображаются в разделе **Проекты**. В нем можно посмотреть информацию о статусе, дате создания, изменения и актуальной версии:



Чтобы создать новый проект:

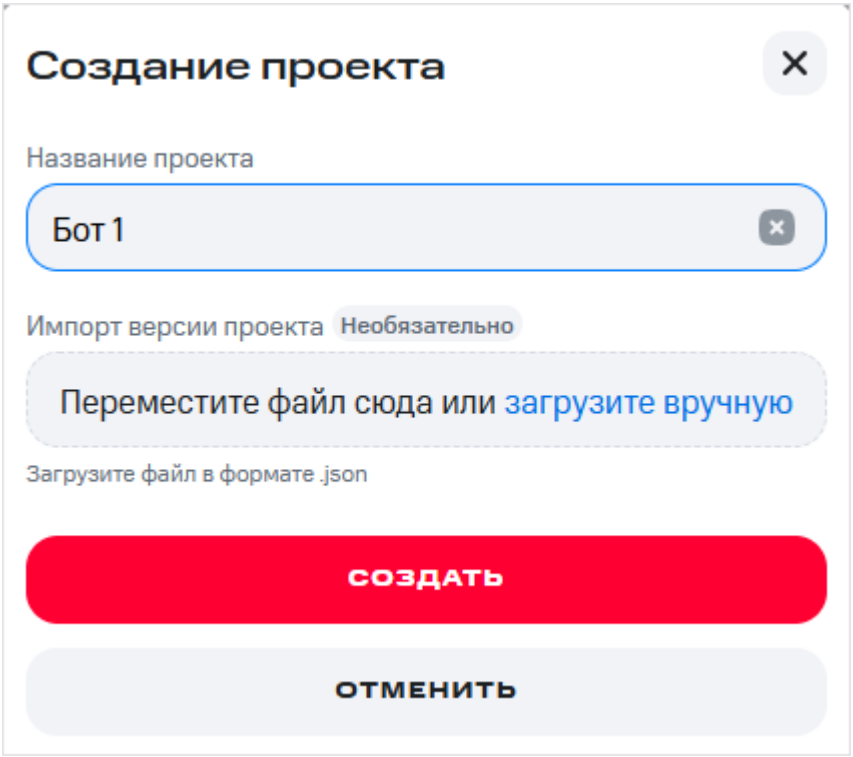
1. Создайте проект.
2. Продумайте структуру сценариев.
3. Разработайте сценарий в конструкторе.
4. Привяжите классификатор для лучшего определения тематики диалога.
5. Добавьте расширенные возможности, например подключите RAG или NER.
6. Создайте новую версию, сохранив состояние текущей.

После этого подготовьте проект к использованию. Если тестирование прошло успешно, опубликуйте версию, чтобы она стала доступна для размещения в канале.

Создание проекта

Чтобы создать проект:

1. На странице **Проекты** нажмите на кнопку **Создать проект**.
2. В открывшемся окне задайте название бота. Затем нажмите на кнопку **Создать**:

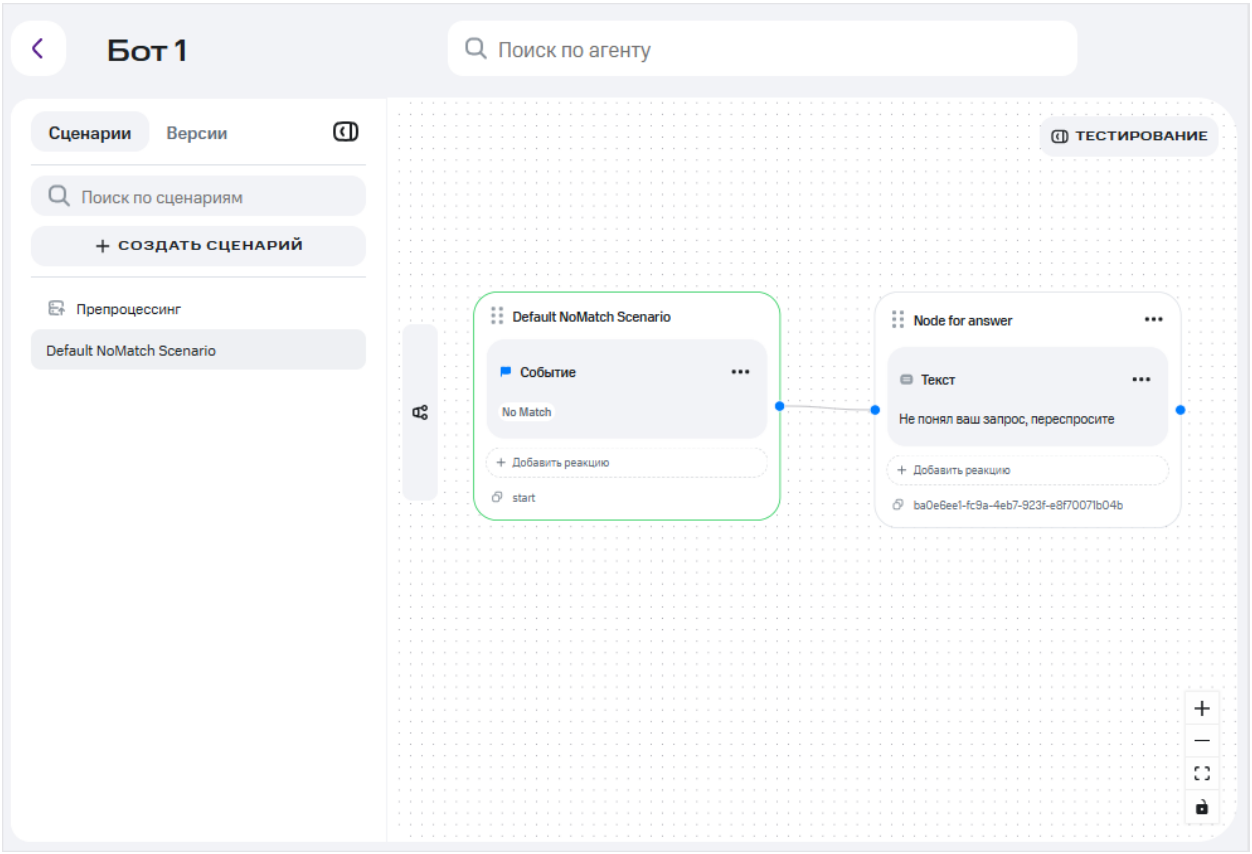



```
:::note

Если JSON-файл проекта ранее был выгружен, то на этом этапе вы можете импортировать
их в систему. Для этого перетащите файл в область загрузки или выберите вручную через
проводник. В результате в созданный проект добавятся сценарии, описанные в файле.

:::
```

В результате открывается конструктор сценариев. Первый сценарий с именем «Default NoMatch Scenario» создается автоматически. Он содержит по одному активационному и реакционному блоку. При необходимости сценарий можно удалить. Также автоматически создается пустой сценарий препроцессинга. Его удаление и изменение имени невозможно.

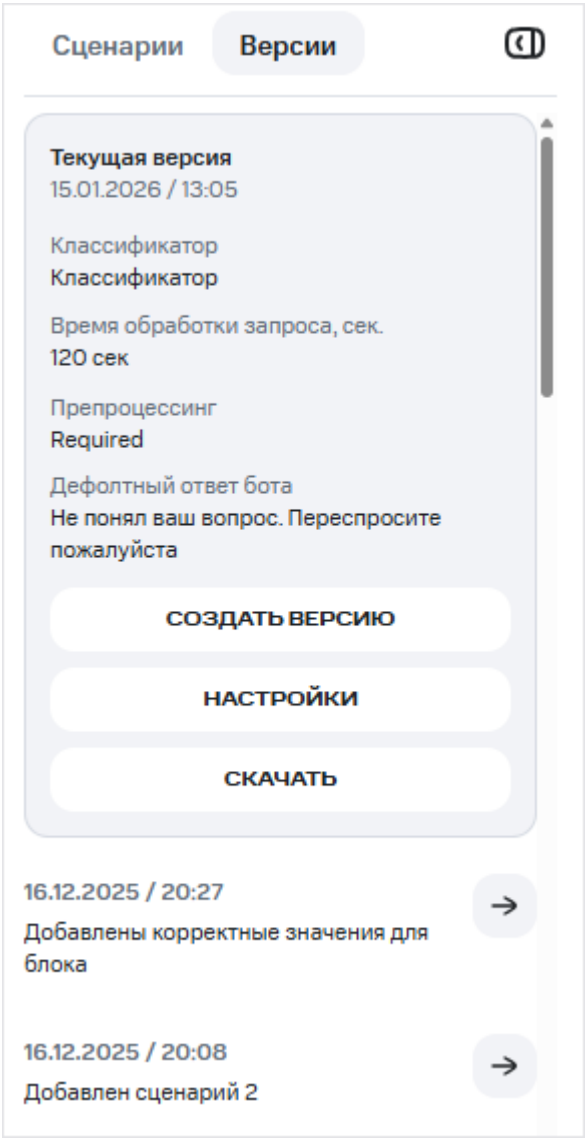


Разработайте все необходимые сценарии, создайте новую версию из текущей и опубликуйте ее.

Версии

Для проектов предусмотрено версионирование. Каждая новая версия проекта подразумевает его усовершенствование. Все изменения вносятся в текущую рабочую версию, по окончании редактирования ее нужно сохранить. После этого версию можно опубликовать, чтобы она стала доступна для выбора в канале.

Сохраненная версия отображается в списке с указанным комментарием. Все новые изменения вносятся в текущую. Предыдущие версии открываются только на чтение.



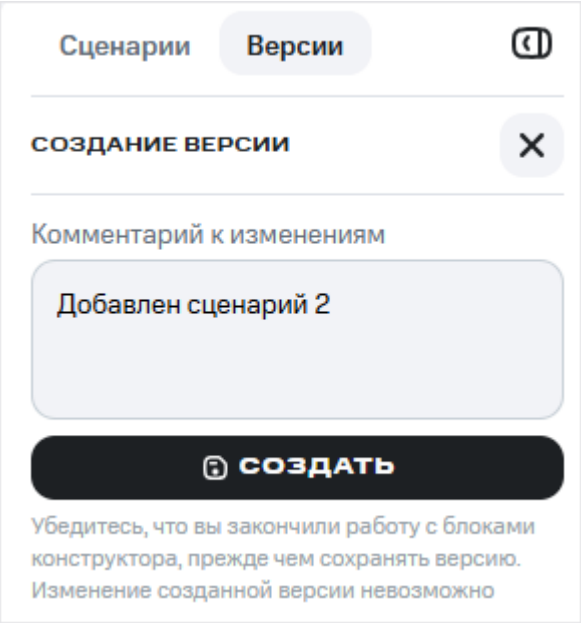
ВНИМАНИЕ

Механизм блокировок работает, если в платформе включена авторизация пользователей.

Изменения в текущей версии сохраняются автоматически каждые 10 секунд бездействия, при переключении на другой сценарий или при выходе из проекта. Благодаря этому изменения не потеряются. При этом виджет тестирования запускается для последнего сохраненного состояния, поэтому рекомендуется предварительно нажать на кнопку **Сохранить изменения**, чтобы учесть последние доработки.

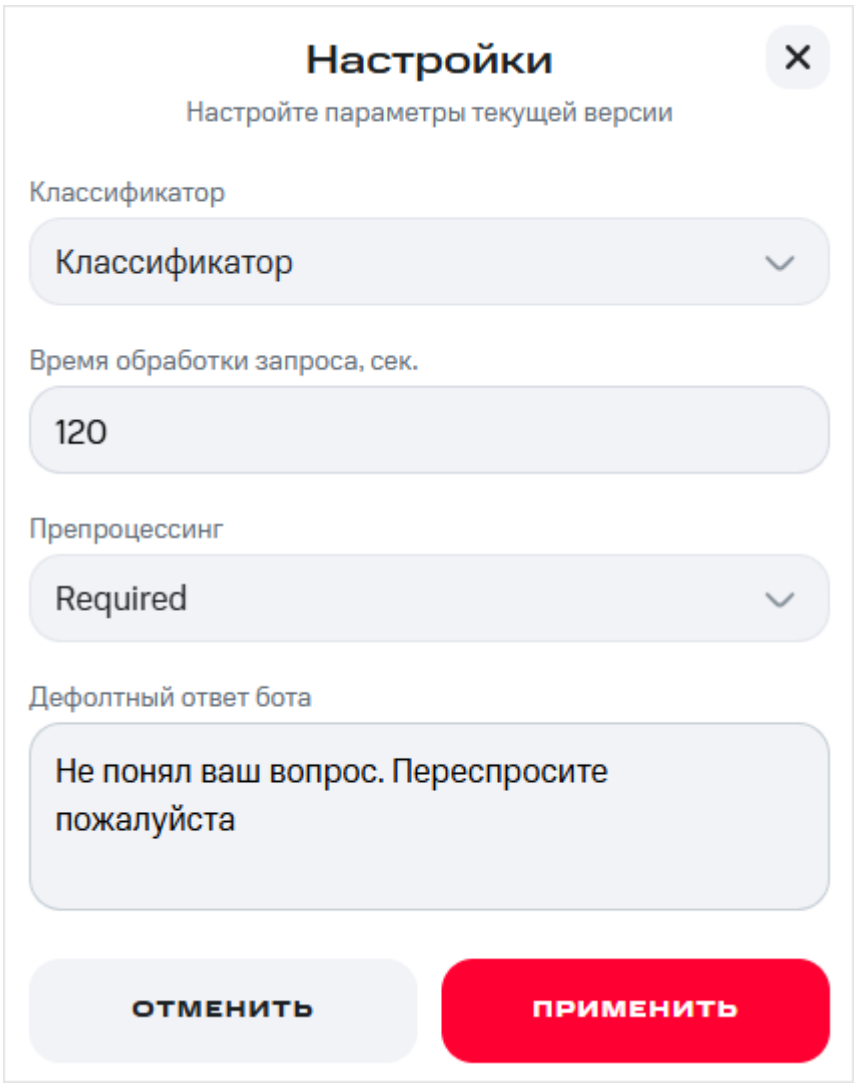
На панели **Версии** для текущей версии доступны кнопки:

- **Создать версию**. Нажмите на кнопку, чтобы создать копию текущей версии и сохранить в ней разработанные сценарии. Укажите комментарий к изменениям и нажмите на кнопку **Создать**.



В результате создается версия и отображается в списке. Дальнейшая работа со сценариями продолжается в текущей версии. Чтобы сохранить внесенные изменения, снова создайте версию;

- **Настройки**. По кнопке открывается окно для изменения настроек версии:



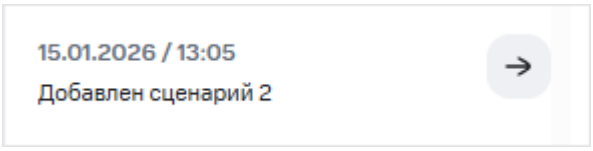
Классификатор. Имя сервиса классификатора. **Время обработки запроса, сек.** Время обработки запроса в секундах. Значение по умолчанию – 5 секунд.

Время обработки запроса должно быть синхронизировано с тем, которое указано в вашем чат-сервисе.

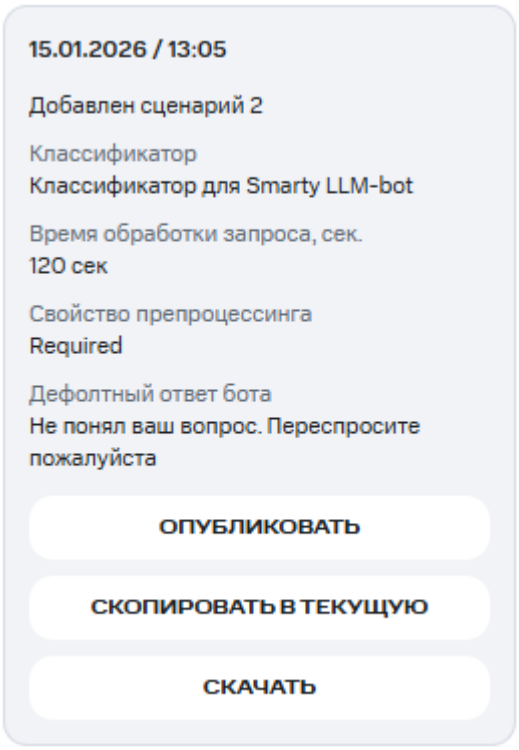
Препроцессинг. Возможные значения: Disabled – препроцессинг отключен, Required – выполнение препроцессинга обязательно. **Дефолтный ответ бота**. Ответ от бота по умолчанию, если стейт не найден.

- **Скачать**. Нажмите на кнопку, чтобы сохранить настройки в формате JSON. При создании нового бота можно импортировать эти настройки.

Чтобы отобразить информацию об одной из предыдущих версий, нажмите на кнопку :



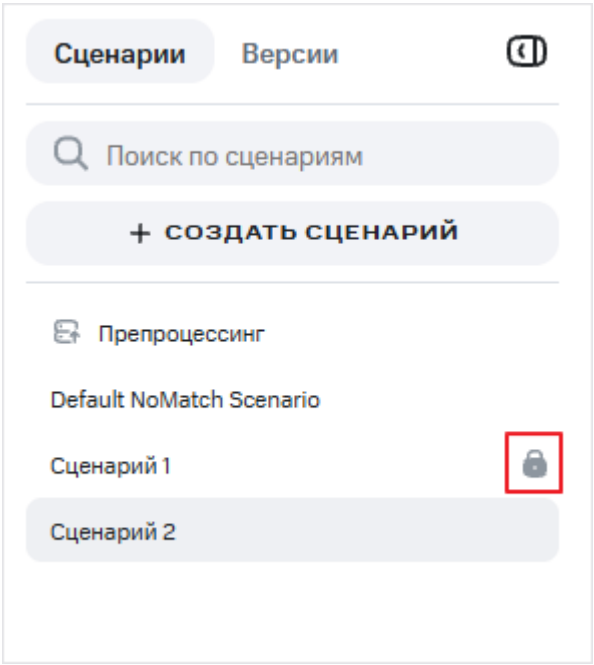
Для предыдущих версий доступны действия:



- **Опубликовать**. Нажмите на кнопку, чтобы опубликовать версию. В списке рядом с опубликованной версией отображается значок ;
- **Скопировать в текущую**. В результате выбранная версия будет скопирована в текущую;
- **Скачать**. Нажмите на кнопку, чтобы сохранить версию в файле формата JSON.

Совместное редактирование

Редактировать текущую версию могут сразу несколько пользователей. При изменении сценария на него устанавливается блокировка. Для других пользователей отображается соответствующий значок:



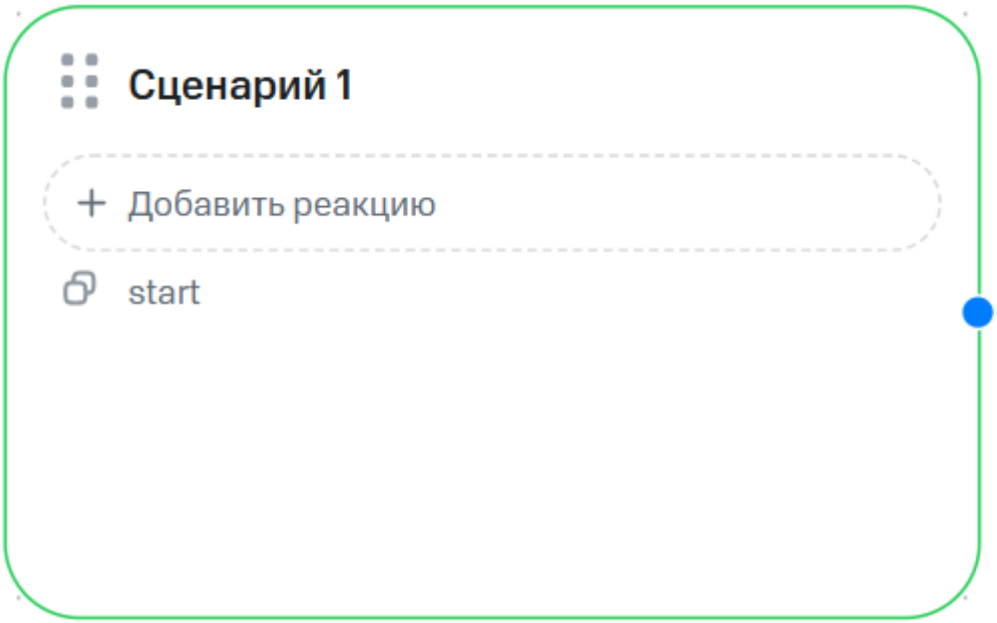
Это означает, что другие пользователи смогут его редактировать, когда:

- текущий пользователь сохранит изменения;
- текущий пользователь перейдет к редактированию другого сценария;
- истечет время бездействия, по умолчанию 5 минут. Кроме этого, для сохранения версии должны быть сняты все блокировки со сценариев.

Создание версии

Чтобы создать новую версию:

1. Создайте сценарий. Для этого на вкладке **Сценарии** нажмите на кнопку **Создать сценарий**. В результате создается новый сценарий с пустой стартовой нодой:



> **ПРИМЕЧАНИЕ**
>
> При создании и редактировании сценария на него устанавливается блокировка. Другие пользователи не могут редактировать его, пока время блокировки не истечет или она не будет снята, например при переходе к редактированию другого сценария.

Задайте имя для сценария. Добавьте в стартовую ноду нужные блоки активации. Наполните сценарий логикой. Для этого добавьте в него реакционные блоки и связи. Подробнее см. раздел «Разработка сценариев».

1. Заполните настройки версии. Для этого нажмите на кнопку **Настройки**. В открывшемся окне заполните поля: **Классификатор**, **Время обработки запроса, сек.**, **Препроцессинг** и **Дефолтный ответ бота**.

> **ВНИМАНИЕ**
>
> Время обработки запроса должно быть синхронизировано с тем, которое указано в вашем чат-сервисе.

1. На панели **Версии** нажмите на кнопку **Создать версию**.

> **ВНИМАНИЕ**
>
> Чтобы сохранить версию при совместном редактировании проекта, нужно дождаться снятия блокировок со всех сценариев.

1. Укажите комментарий к изменениям. Нажмите на кнопку **Создать**.

В результате создается копия текущей версии и сохраняется с указанным комментарием в списке версий. Чтобы она стала доступна в канале, ее нужно опубликовать. Все новые изменения вносятся в текущую версию. При необходимости вы можете вернуть состояние ранее сохраненной версии. Для этого перейдите в нее и нажмите на кнопку **Скопировать в текущую**. Сценарии и настройки этой версии будут скопированы в рабочую.

(пусто)

Создание сценария

В качестве запускающего механизма для сценария используются блоки активации. Они определяют, когда бот должен активироваться и начать взаимодействие с пользователем. Для каждого блока активации создаются блоки реакции – они определяют, как бот должен реагировать на входящее сообщение.

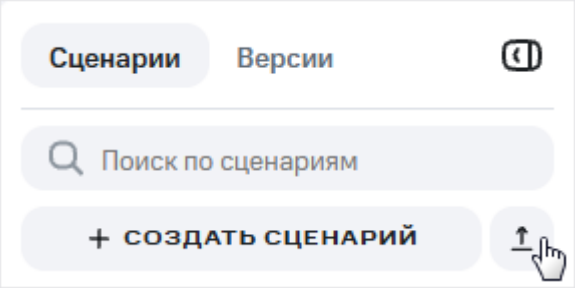
Чтобы создать новый сценарий:

1. Продумайте структуру.
2. В конструкторе сценариев на вкладке **Сценарии** нажмите на кнопку **Создать сценарий**.
3. Кликом правой кнопки мыши по созданному сценарию вызовите контекстное меню. В нем выберите пункт **Переименовать**. Заполните название сценария и нажмите ENTER.

ВНИМАНИЕ

В имени допускаются только буквы, цифры, пробелы, дефис и нижнее подчеркивание.

1. Наполните сценарий заранее разработанной логикой. Для этого добавьте нужные блоки и связи между ними. При необходимости вы можете импортировать в проект сценарий в формате JSON. Для этого нажмите на кнопку импорта сценария:



ПРИМЕЧАНИЕ

В реакционный блок можно добавлять сразу несколько блоков. В этом случае он является нодой и все блоки внутри него выполняются последовательно.

1. Создайте все необходимые сценарии аналогичным образом.
2. Привяжите классификатор. Классификатор позволяет группировать ответы пользователей на основе предварительно размеченных данных. Это означает, что можно задать фразу или набор фраз, которые послужат эталоном для сравнения с запросом пользователя. Эти запросы будут проверяться на семантическое соответствие заданным фразам. Если порог такого соответствия достаточно высок, то можно считать, что две реплики относятся к одному и тому же классу. Соответственно, чат-бот будет на них реагировать одинаково.

Создание структуры

Перед созданием сценариев продумайте их логическую структуру. Сценарии должны определять логику работы бота. Они описывают переходы бота из одного состояния в другое в зависимости от полученного ответа клиента. Важно продумать максимальное количество возможных вариантов запросов и ответов, а также описать переход в блок с типом No match, когда для запроса не удалось найти подходящее условие активации.

Чтобы создать логическую структуру:

1. Определите тематики запросов, которые можно сгруппировать, и используйте их при именовании сценариев.

   Например, бот для обслуживания клиентов медицинской организации может записать клиента к врачу, зарегистрировать в программе лояльности или показать информацию о доступных услугах. В этом случае нужно создать три сценария: «Запись к врачу», «Регистрация в программе лояльности», «Справка по услугам».

2. Продумайте логику активации. Определите варианты ситуаций и реакции на них.

   Например, пользователь отправляет боту сообщение: «Привет». В этом случае:

   - блок активации бота изменяется, бот активируется для начала диалога;
   - блок реакции определяет, что бот должен ответить пользователю приветствием, например: «Привет! Как дела?»

После разработки структуры приступайте к созданию сценария в конструкторе.

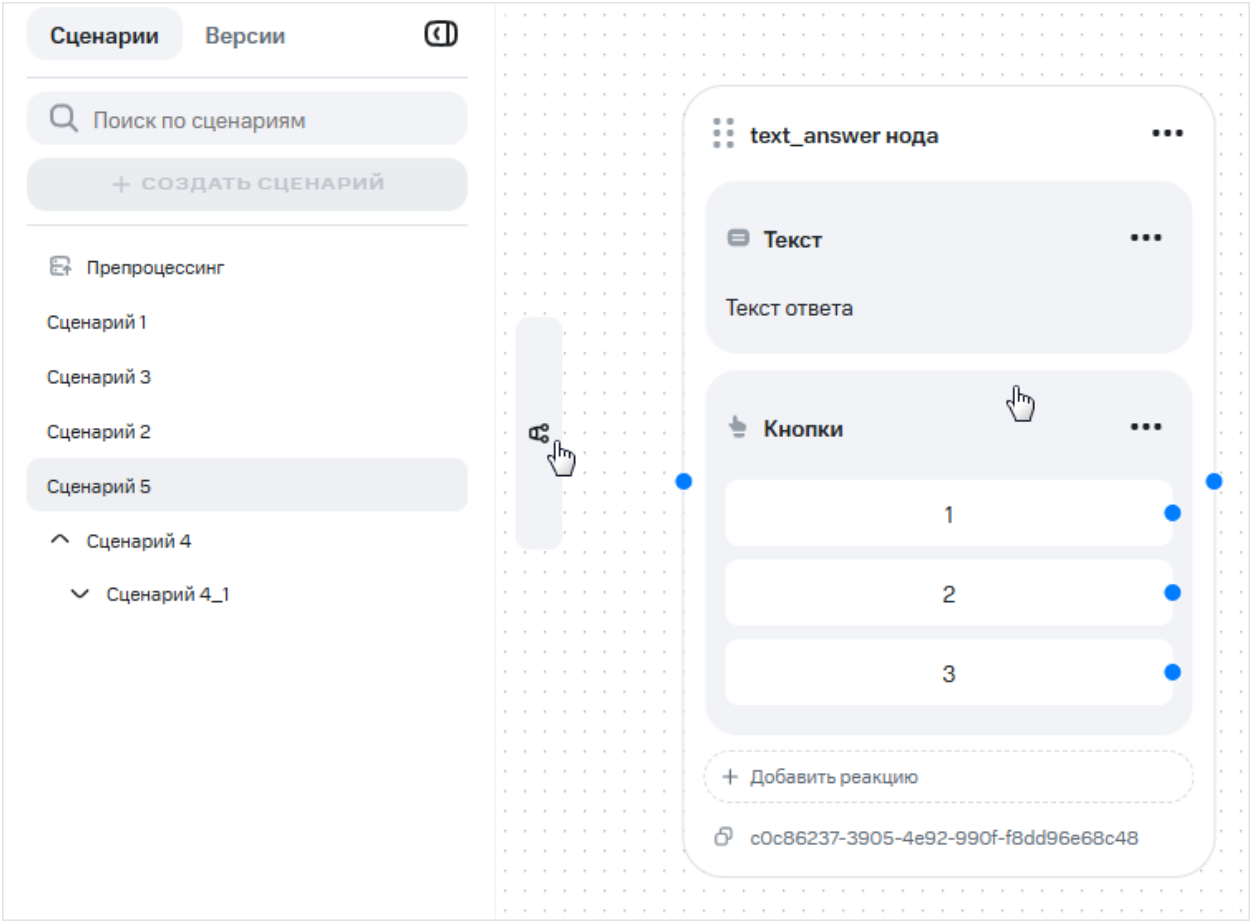
Работа в конструкторе

Сценарий представляет собой последовательность связанных нод, внутри которых располагаются блоки. В зависимости от действий пользователя срабатывает определенная реакция.

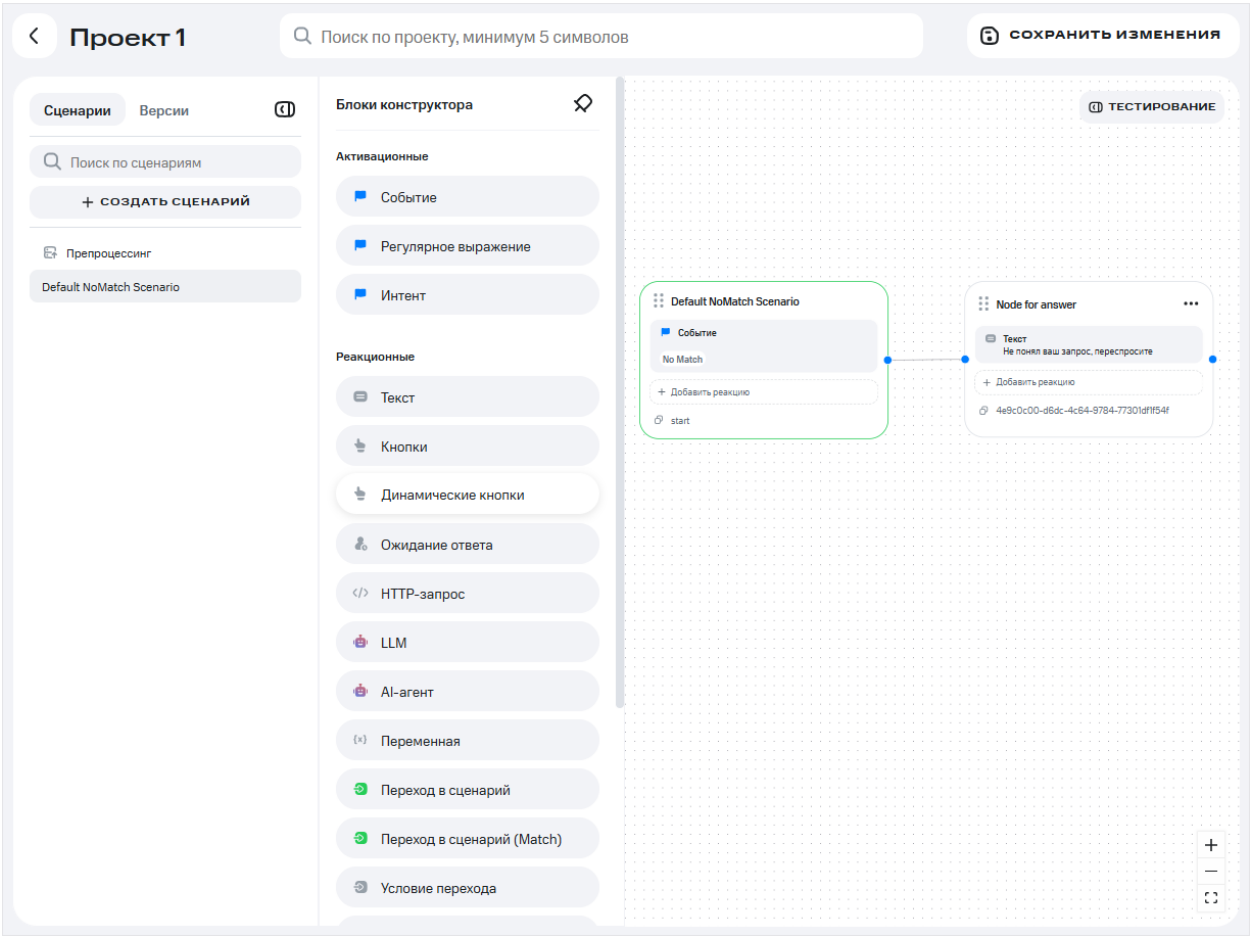
Ноды состоят из блоков – это элементы, которые описывают, как сработает определенный сценарий и какие действия при этом выполняются. Связи представлены в виде линий, показывающих переходы.

Для одного бота можно добавить несколько сценариев, каждый из которых может содержать вложенные. Список сценариев, блоки и настройки версии доступны в левой части конструктора. При необходимости можно настроить переход из одного сценария в другой.

Чтобы отобразить список доступных блоков, нажмите на кнопку :



В результате открывается панель Блоки конструктора. Для удобства ее можно закрепить по кнопке .



Рабочая область

Вся работа со сценариями ведется в рабочей области. Она представляет собой двумерное пространство, которое не ограничено в высоту и ширину.

Чтобы добавить элементы, перетащите их с панели компонентов в рабочую область (drag-and-drop). Изменить масштаб поля можно прокруткой колесика мыши.

Также в рабочей области доступен поиск по элементам сценария. Например, введите в строке поиска название блока, и фокус переместится на первый найденный блок с совпадением.

На рабочей области доступны действия:

КНОПКА	ОПИСАНИЕ
+	Увеличить масштаб
—	Уменьшить масштаб
⛶	Изменить масштаб, чтобы сценарий поместился в зону видимости
🔓	Установить/снять блокировку на сценарий

Сценарии

Работа со сценариями ведется во вкладке **Сценарии** на панели в левой части экрана.

По умолчанию в новом проекте автоматически создаются сценарии:

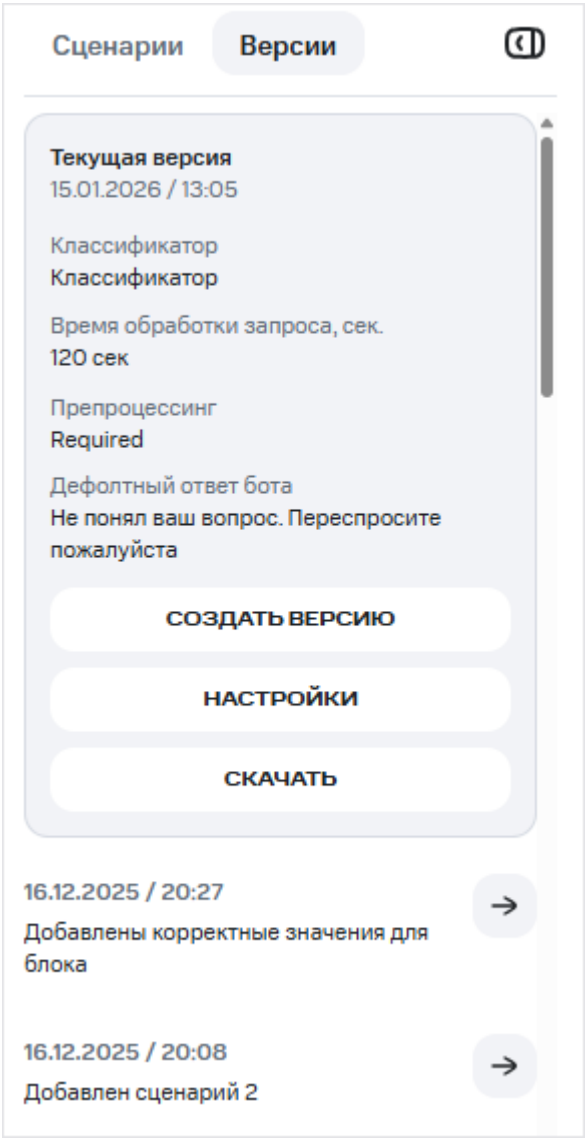
- Default NoMatch Scenario – для ситуаций, когда интент или регулярное выражение не определены;
- Препроцессинг – это технический сценарий, который выполняется сразу после поступления запроса от пользователя и до выбора блока активации.

Чтобы добавить новый сценарий, нажмите на кнопку **Создать сценарий**. По щелчку правой клавиши мыши на имя сценария доступно контекстное меню, из которого можно создать вложенные сценарии (доступно после сохранения версии), переименовать, дублировать, удалить сценарий или скопировать его slug (слаг) – уникальный ключ, который представляет собой имя сценария строчными буквами. В качестве разделителя вместо пробела используется знак нижнего подчеркивания. Slug предназначен для задания логики перехода в блоке **Скрипт**.

> **ВНИМАНИЕ**
>
> При необходимости обратиться к сценарию рекомендуется использовать slug сценария вместо его идентификатора, так как он остается постоянным, а ИД может измениться, например при импорте сценария.

Версии

Для работы с версиями используется вкладка **Версии**. На ней отображаются все созданные версии и информация о них.

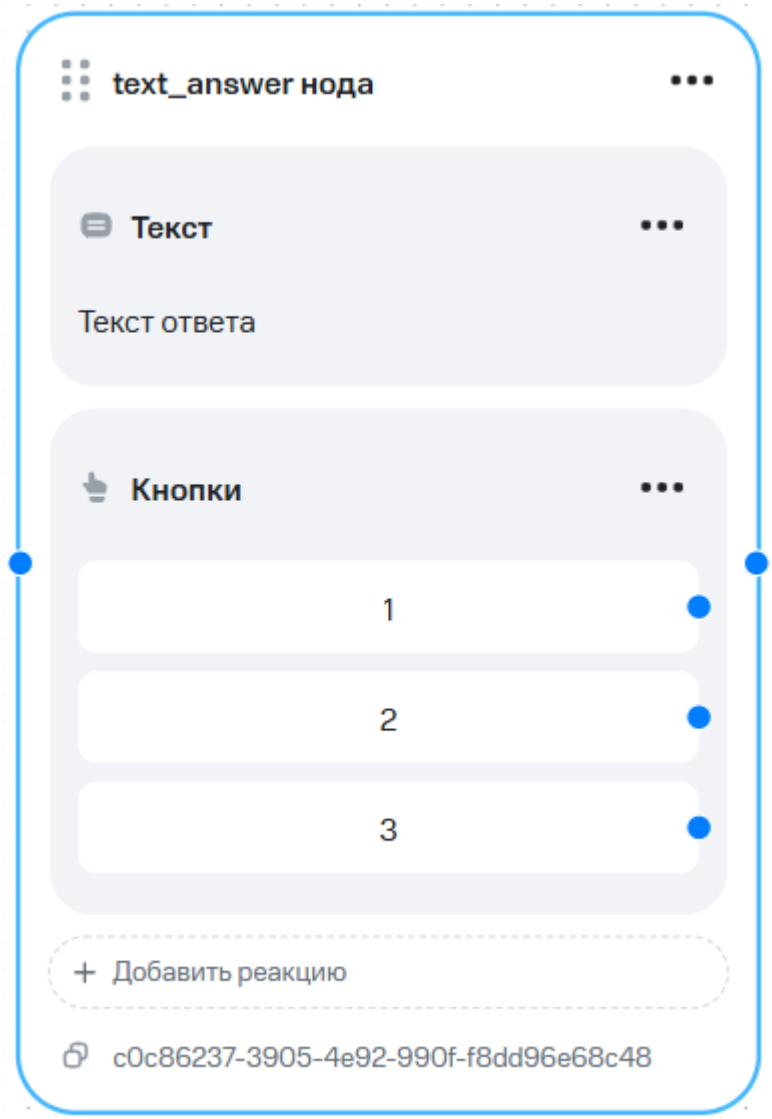


Подробнее см. раздел «Версии».

Блоки конструктора

Сценарии состоят из нод. Нода представляет собой блок, внутри которого содержится набор блоков. Они выполняются последовательно в рамках одного стейта. В конструкторе сценариев доступны активационные и реакционные блоки. Блоки активации определяют, по каким действиям пользователя бот переходит к конкретному сценарию. При попадании в этот сценарий выполняются блоки реакции.

Доступные для добавления блоки располагаются на панели **Блоки конструктора**. Чтобы добавить блок в нужную ноду, перетащите его с помощью левой клавиши мыши.



По кнопке ••• ноду можно переименовать или удалить.

В нижней части нод и блоков располагаются идентификаторы. При необходимости их можно скопировать. Для этого нажмите на кнопку . Например, при редактировании большого сценария запустите поиск по ИД, чтобы вернуться к заполнению блока.

Для удобства используйте горячие клавиши:

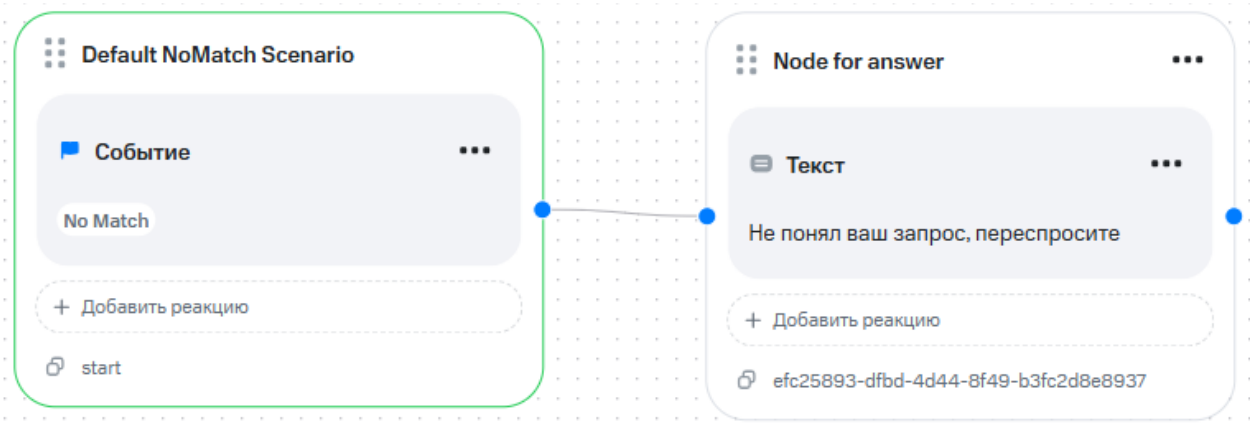
- CTRL+C – копировать блок;
- CTRL+V – вставить скопированный блок;
- CTRL+Z – отменить действие;
- DELETE/BACKSPACE + выделенный блок/нода/связь – удалить выделенный элемент.

ВНИМАНИЕ

В сценарии рекомендуется использовать не более 200 блоков, так как большое количество блоков может повлиять на производительность.

Связи

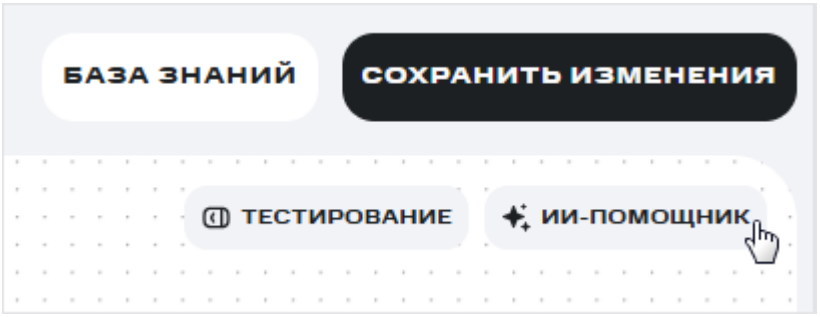
Связи между нодами образуют последовательность выполнения сценария. Чтобы создать связь, нажмите на синюю метку на блоке и протяните курсор мыши к нужному блоку.



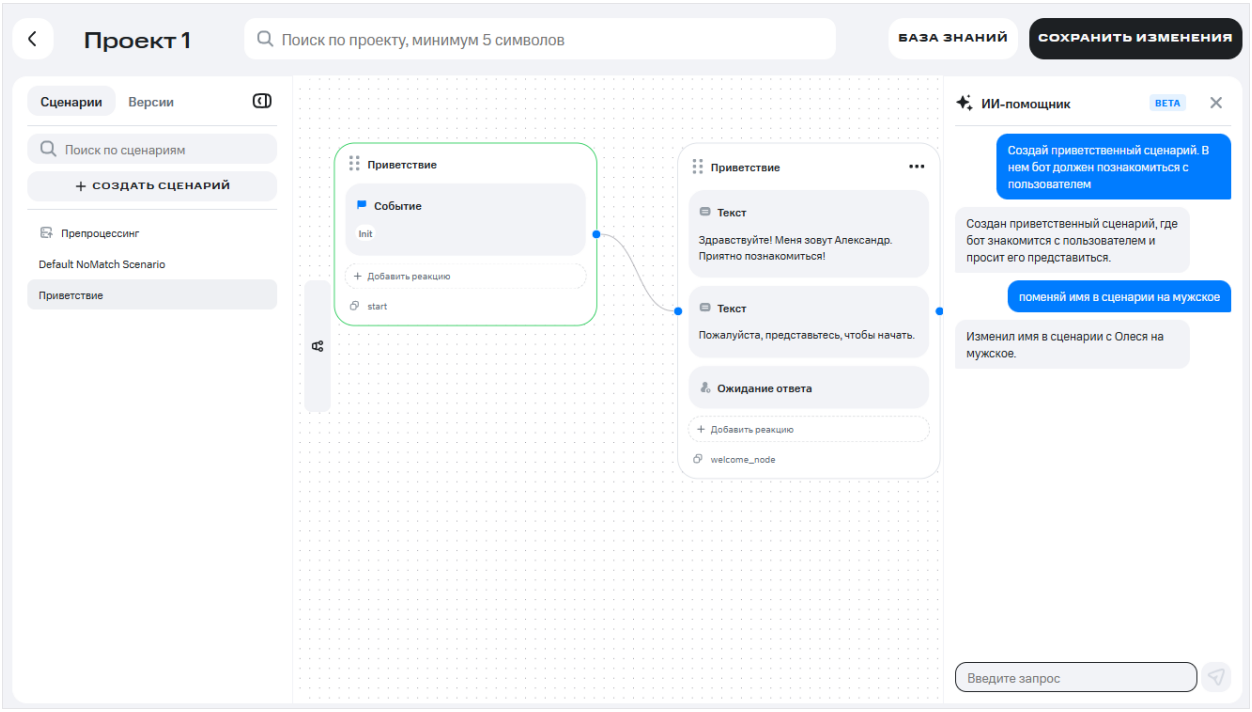
Чтобы выполнять блоки последовательно в рамках одной ноды, добавьте их друг за другом, перетащив из панели компонентов в область **Добавить реакцию**.

Использование ИИ-помощника

В MWS AI Agents Platform вы можете создать базовые сценарии в пару кликов. Для этого в конструкторе сценариев доступен интеллектуальный помощник:



Нажмите на кнопку **ИИ-помощник**, чтобы открыть чат. Введите требования к сценарию и дождитесь ответа:



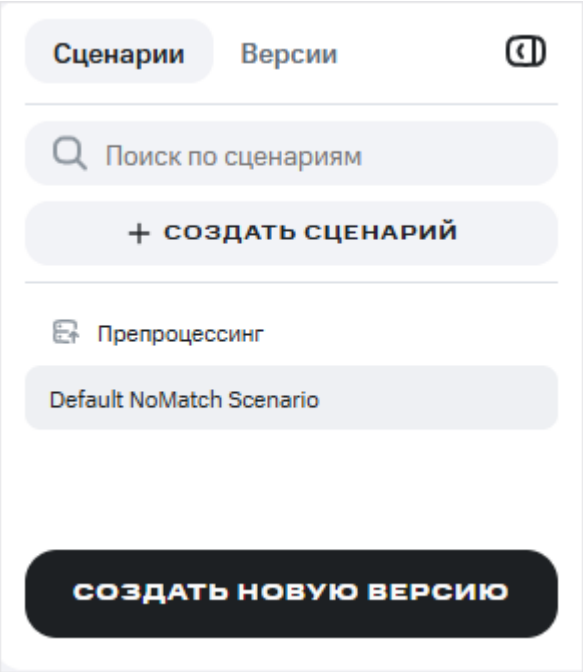
Вы можете изменить или доработать сгенерированные сценарии вручную, а также попросить ИИ-помощника исправить их самостоятельно.

(пусто)

Компоненты

При создании проекта автоматически создаются сценарии:

- Препроцессинг – это технический сценарий, который используется для обогащения контекста сессии. Позволяет повысить качество и скорость обработки запроса пользователя. Выполняется сразу после поступления запроса от пользователя и до выбора блока активации. По умолчанию не содержит блоков;
- Default NoMatch Scenario – сценарий по умолчанию. Содержит блок активации **Событие** с типом **No Match** и блок реакции **Текст**. Остальные сценарии нужно создавать вручную.



При поступлении запроса от пользователя сначала выполняется препроцессинг, если в настройках версии указано его выполнение. После этого проверяются активационные блоки остальных сценариев. Как только срабатывает блок активации, выполняются реакционные блоки соответствующего сценария.

На панели **Блоки конструктора** располагаются доступные активационные и реакционные блоки. Кроме этого, в разделе Сохраненные отображаются блоки, которые были заполнены параметрами и сохранены для повторного использования. Чтобы добавить блок, наведите курсор мыши на компонент и, удерживая левую клавишу мыши, перетащите его в рабочую область. После создания блок можно перемещать по рабочей области с помощью механизма drag-and-drop.

> **ВНИМАНИЕ**
>
> Состав доступных для добавления блоков зависит от типа сценария. Для сценария «Препроцессинг» набор блоков ограничен.

Для всех активационных и реакционных блоков можно заполнить поле **Тэги** по кнопке **Добавить тэги**. Примеры: zapis_yes, good_bot, success. Значения из этого поля используются для потребностей аналитики диалогов, например для расчета метрик. Кроме этого, при заполнении блоков можно указывать зарезервированные переменные.

Препроцессинг

До активации основных сценариев в боте может выполняться препроцессинг. Для одной версии бота допускается один такой сценарий. Выполняется после каждого запроса пользователя. Препроцессинг не может содержать блоки для взаимодействия с пользователем в чате и перевод на оператора. При этом в препроцессинге можно задать условие прямого перехода в конкретный сценарий бота, определить значения переменных, использовать интеграционные функции HTTP-вызова и т.д.

ВНИМАНИЕ

Чтобы сценарий препроцессинга выполнялся, в настройках версии в поле **Препроцессинг** должно быть установлено значение **Required**.

Если в препроцессинге используется блок **Переход в сценарий** и в нем установлен флажок **Вернуться после завершения**, то сценарий, к которому нужно перейти, также выполняется в рамках препроцессинга. Поэтому проверьте, что в нем используются только допустимые блоки: **HTTP-запрос**, **Переменная**, **Переход в сценарий**, **Переход в сценарий (Match)**, **Условие**, **Скрипт**. При попытке выполнить недопустимые блоки возникает ошибка.

Кроме этого, в сценарии препроцессинга задается вычисление кандидатов для активации по интентам.

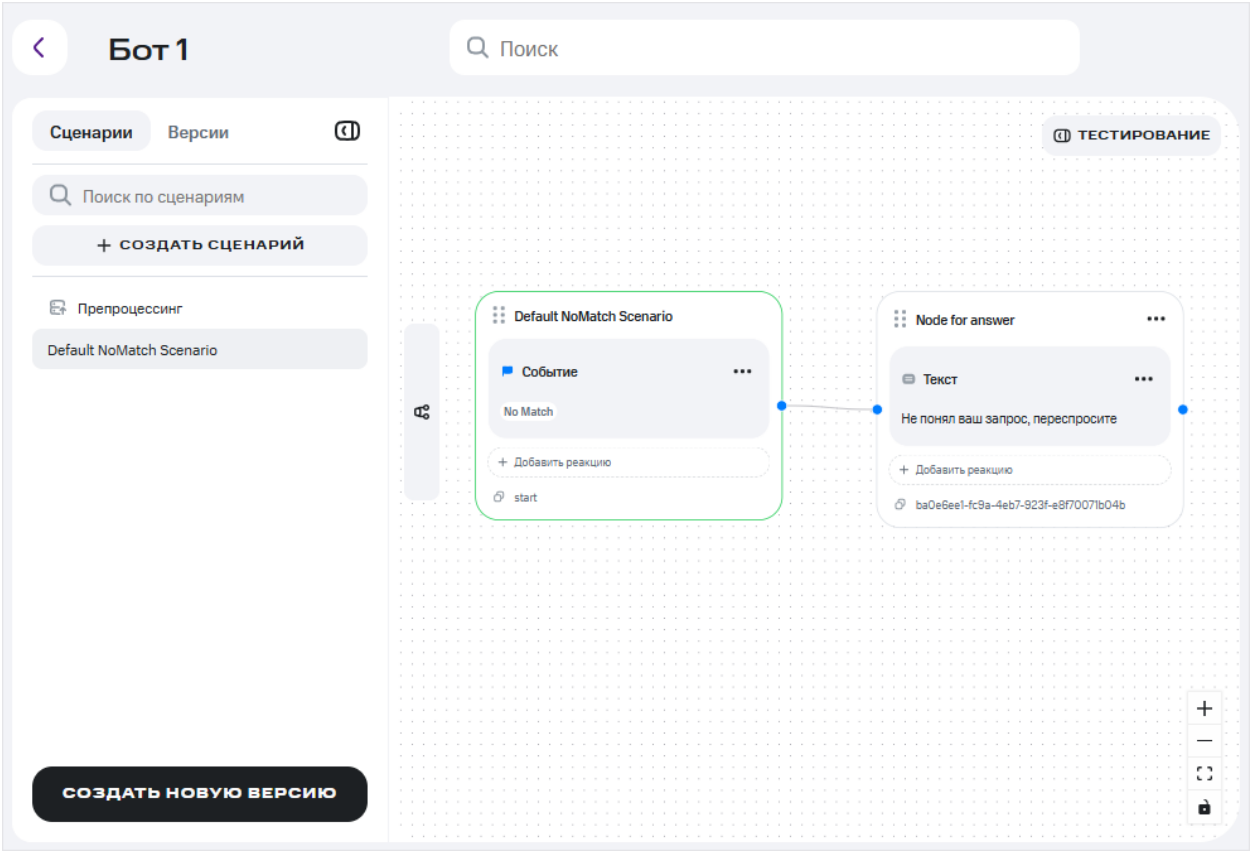
Подробнее об активации см. раздел «Блоки активации».

Информация о работе этого типа сценария записывается в историю диалогов аналогично другим сценариям.

Препроцессинг нельзя удалить. Если он не нужен для исполнения логики вашего бота, то оставьте его пустым.

Default NoMatch Scenario

При создании бота по умолчанию создается сценарий, в который бот попадает, если интент или регулярное выражение не определены:



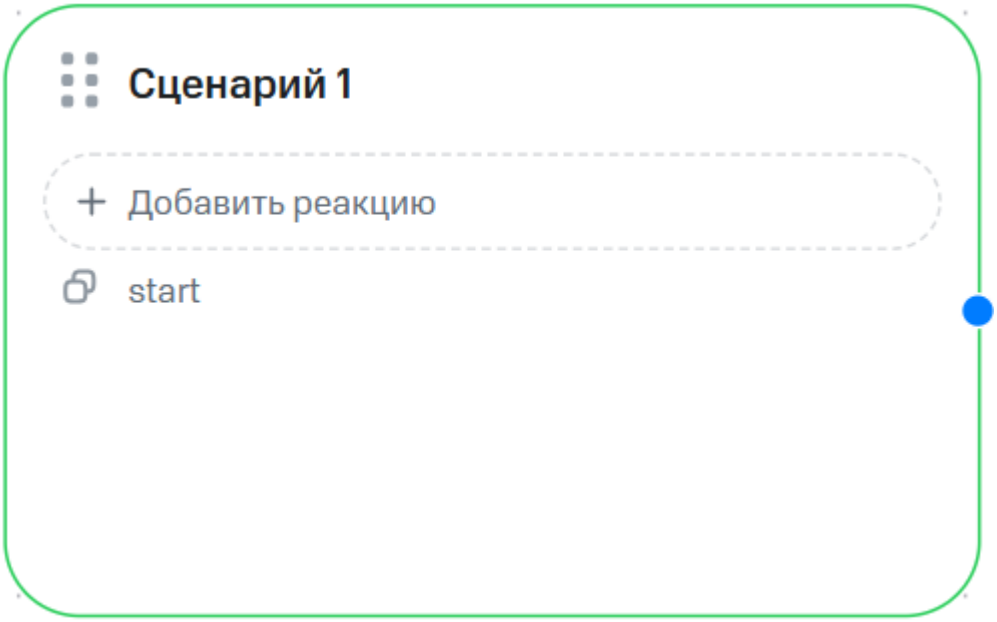
При необходимости сценарий можно удалить.

(пусто)

Использование блоков активации

Блоки активации определяют, по какому запросу пользователя бот переходит к выполнению конкретного сценария. Сценарий может выбираться по одному из условий активации:

- событие (Event);
- интент (Intent);
- регулярное выражение (Match). Блоки активации можно добавлять только в стартовый стейт:

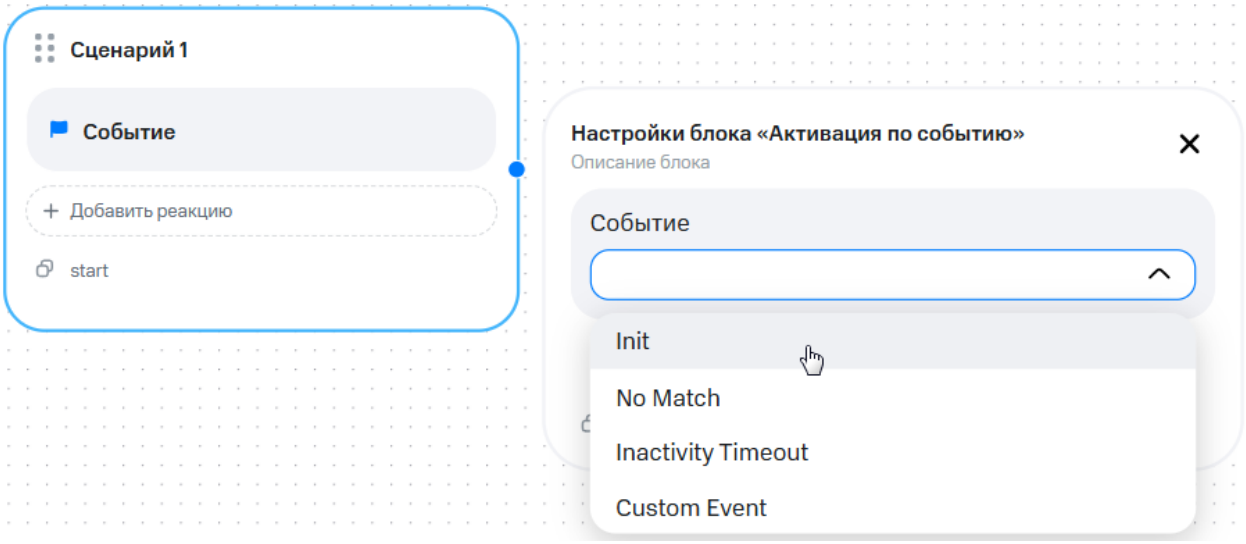


> **ПРИМЕЧАНИЕ**
>
> Приоритет активации настраивается в параметре **PRIORITY_ACTIVATOR** файла values.yaml.

Событие

Активация по событию:

- по первому запросу пользователя;
- в случае, когда не удалось определить сценарий для активации;
- по таймауту бездействия;
- по событию, пришедшему с поверхности.



ВНИМАНИЕ

Активация по пользовательскому событию поддерживается только в проектах, подключенных к поверхности через канал HTTP. Если используется другой тип канала, то активация по пользовательскому событию не выполнится.

В выпадающем списке выберите одно из возможных значений:

- **Init** – событие начала диалога. Сценарий активируется, если у пользователя не было открытой сессии до обработки текущего сообщения. Если в блоке активации по событию указано значение **Init**, то при первом же сообщении от пользователя выбирается сценарий, в котором находится этот блок;
- **No Match** – не удалось определить сценарий для активации;
- **Inactivity Timeout** – истекло время бездействия пользователя. Сценарий активируется, если пользователь неактивен в течение некоторого времени. Время задается в настройках канала в параметре **Таймаут бездействия**. При этом указанный таймаут должен быть меньше времени жизни диалога. Если таймаут указан, а сценария с таким блоком активации нет, то значение параметра игнорируется. Подробнее о настройках канала см. в разделе «Каналы».
- **Custom Event** – пользовательское событие. Тип предназначен для активации по событию, пришедшему с поверхности по протоколу HTTP. Примеры событий: открытие виджета с чатом на сайте, комментарий к посту в социальной сети, поступление входящего сообщения на почтовый адрес компании и т.д. Событие отслеживается на поверхности, его имя и другая информация поступает в запросе к HTTP-адаптеру. Подробнее о структуре запроса см. раздел «Метод POST /api/{channelId} – получить входящее сообщение от http-клиента».

ПРИМЕЧАНИЕ

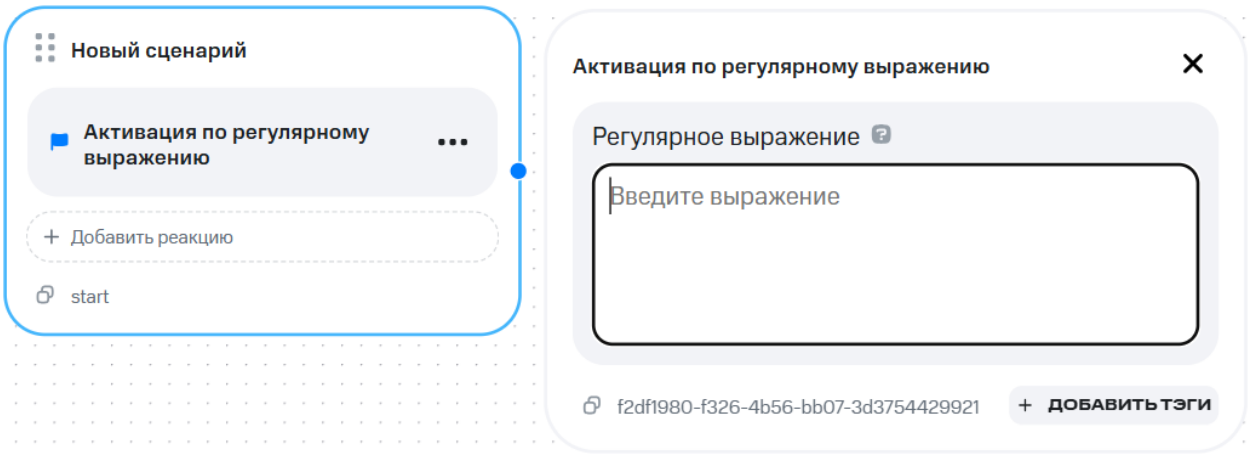
Чтобы протестировать активацию по пользовательскому событию, в тестовом виджете переопределите системную переменную **system.input**: задайте для нее тип и имя события, которые нужно проверить.

Пример настройки контекста:

```
{
  "system": {
    "surfaceMetadata": {},
    "input":
    {
        "type": "event",
        "name": "my_event",
        "data": {
            "data_1":"data_1_value"
         }
    }
  },
  "session": {},
  "request": {},
  "temp": {}
}
```

Регулярное выражение

Активация по регулярному выражению. Для активации стейта происходит поиск соответствия запроса пользователя регулярному выражению. Укажите его в поле **Регулярное выражение**.

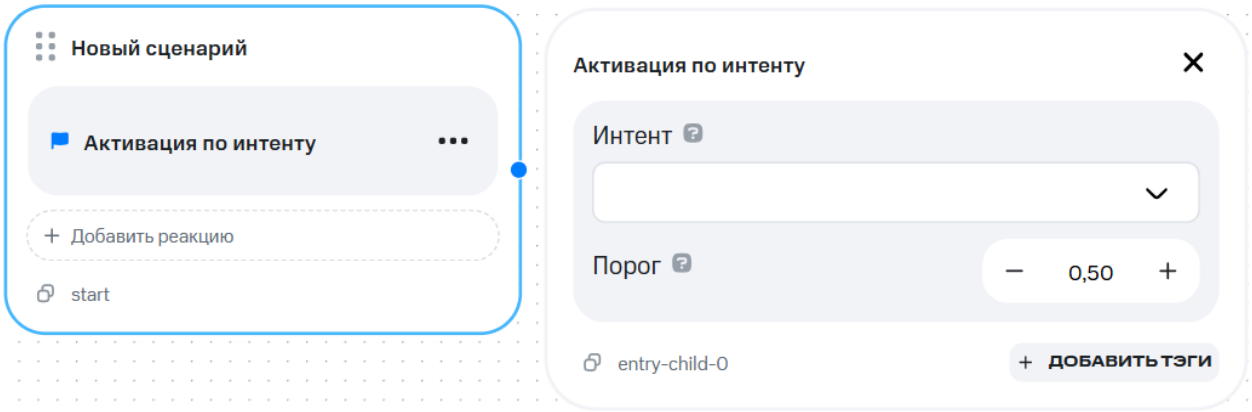


ВНИМАНИЕ

При написании регулярного выражения используйте стандартный синтаксис regexp.

Интент

> **ВНИМАНИЕ**
>
> Кандидаты-интенты для активации не вычисляются автоматически. Чтобы получить интенты для активации, в сценарии препроцессинга напишите скрипт для определения кандидатов-интентов.



Активация по интенту. Чтобы активировать стейт, выполняется запрос к классификатору для определения интента по запросу пользователя.

> **ВНИМАНИЕ**
>
> Имя классификатора указывается на вкладке с настройками версии в поле **Классификатор**. Интент должен соответствовать одному из существующих интентов в подключенном классификаторе.

Заполните поля:

- **Интент** – название интента;
- **Порог** – значение, при достижении которого совпадающий со значением в данном блоке интент, полученный от классификатора, рассматривается в качестве активатора сценария. Значение сравнивается с параметром **score**, которое также возвращает классификатор вместе с интентом. По умолчанию 0,5.

Чтобы активационный блок **Интент** сработал, предварительно выполните:

1. Убедитесь, что для версии бота включено использование препроцессинга.
2. В сценарий препроцессинга добавьте блок Скрипт.
3. В асинхронной функции handler(context: Context) с помощью предопределенной функции predict_intent получите наиболее вероятные интенты для активации запроса.

   Формат функции:

   ```
   context.nlu.predict_intent(message, top_n=1)
   ```

   Где: **message** – сообщение, для которого нужно определить наиболее подходящих кандидатов; **top_n** – количество наиболее вероятных кандидатов. Например, в ситуациях, когда одному интенту соответствует сразу несколько блоков активации в разных сценариях, можно указать top_n=1. Это

обеспечит наличие только одного подходящего кандидата в списке наиболее вероятных. Полученные интенты сохраняются в виде массива в переменную **context.nlu.intents**. Элементы массива упорядочены по вероятности и по приоритету иерархии. В переменной context.nlu.raw_intents формируется массив из top_n интентов, состоящий из имен интентов и коэффициентов вероятности (score). Для обратной совместимости имя первого интента из массива **context.nlu.raw_intents** и его score сохраняются в переменные **context.system.sure_topic** и **context.system.topic_score** соответственно.

В результате сработает активация по интенту. При необходимости переопределите кандидатов вручную.

Пример скрипта:

```
async def handler(context: Context) -> None:
    message = context.system.last_user_message.strip()
    # Получение top_n кандидатов для активации по интенту
    await context.nlu.predict_intent(message, top_n=1)

    # Ручное переопределение списка кандидатов для активации по регулярному
выражению
    if message == "message":
        context.nlu.intents = []
        context.nlu.matches = [
            ActivationCandidate(
                scenario_id=1,
                edge=MatchEdge(value="scenario_1",
target_node_id="c34dcad6-5343-4f06-a55d-bf68424b28b1"),
                score=1.0,
            )
        ]
```

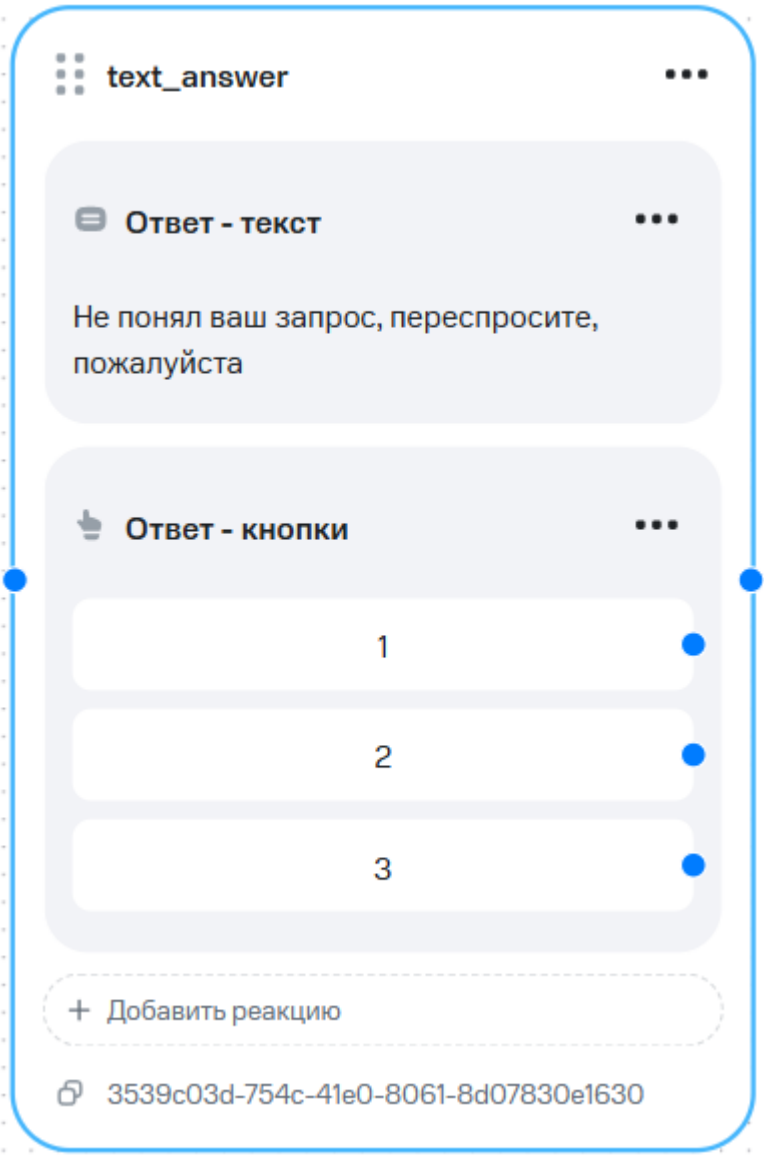
(пусто)

Использование блоков реакции

После того, как срабатывает активационный блок в сценарии, выполняются блоки реакции. Например, бот может отправить пользователю текст, кнопки для выбора вариантов ответов или автоматически перейти к другому сценарию.

Для настройки логики можно использовать зарезервированные переменные, а также переменные, созданные в сценарии. При этом в качестве префикса указывайте для них области видимости.

Чтобы добавить блок в рабочую область, перенесите его с панели **Блоки конструктора**. При этом в один блок (стейт, ноду) можно добавить несколько реакционных блоков. Блоки будут выполняться последовательно.



Для сценария препроцессинга набор блоков ограничен.

ПРИМЕЧАНИЕ

При заполнении блоков вы можете задействовать функцию **normalize**, которая выполняет лемматизацию слов – преобразует слова в их начальные, базовые формы (леммы).

Формат заполнения: {{normalize(Имя переменной)}}

Примеры лемматизации слов:

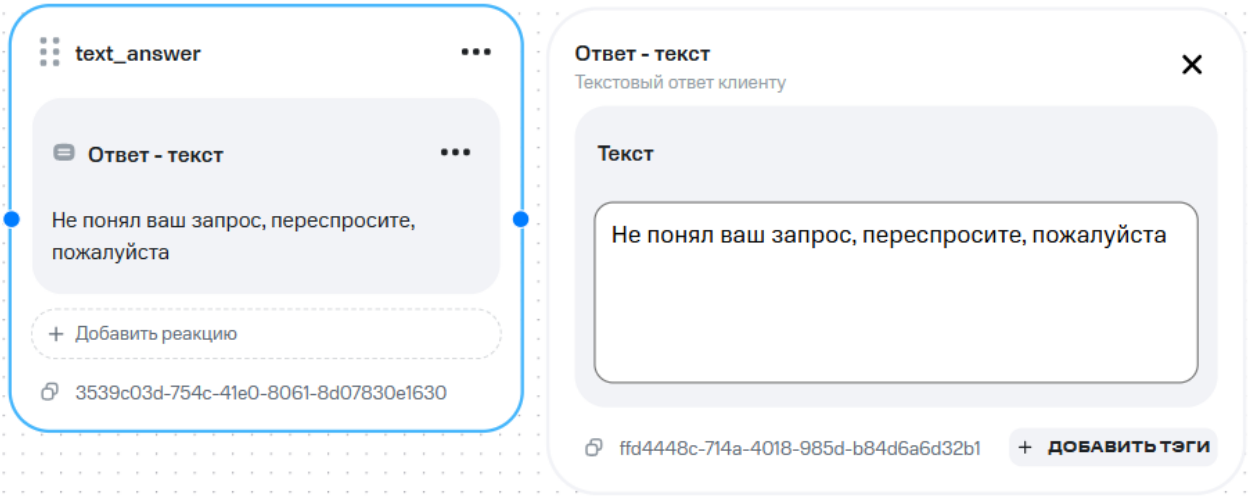
- «бежал», «бежит», «бегущий» – «бежать»;
- «стола», «столов», «столами» – «стол».

Например, в сообщении клиента передано местоположение – «к неврологу». Чтобы сформировать название кнопки, требуется наименование города в начальной форме – «невролог». Для этого в текстовом поле укажите переменную **last_user_message** с добавлением функции **normalize**: {{normalize(system.last_user_message)}}. Результатом работы блока будет кнопка с лемматизированным сообщением клиента – «невролог». Учитывайте, что в начальные формы возвращаются все слова в сообщении.

При лемматизации регистр всех букв в тексте меняется на нижний.

Текст

Текстовый ответ клиенту.



Заполните поле **Текст** – ответ, который нужно выводить в чате с пользователем. При необходимости в поле Текст можно использовать переменные, заключенные в двойные фигурные скобки **{{}}**.

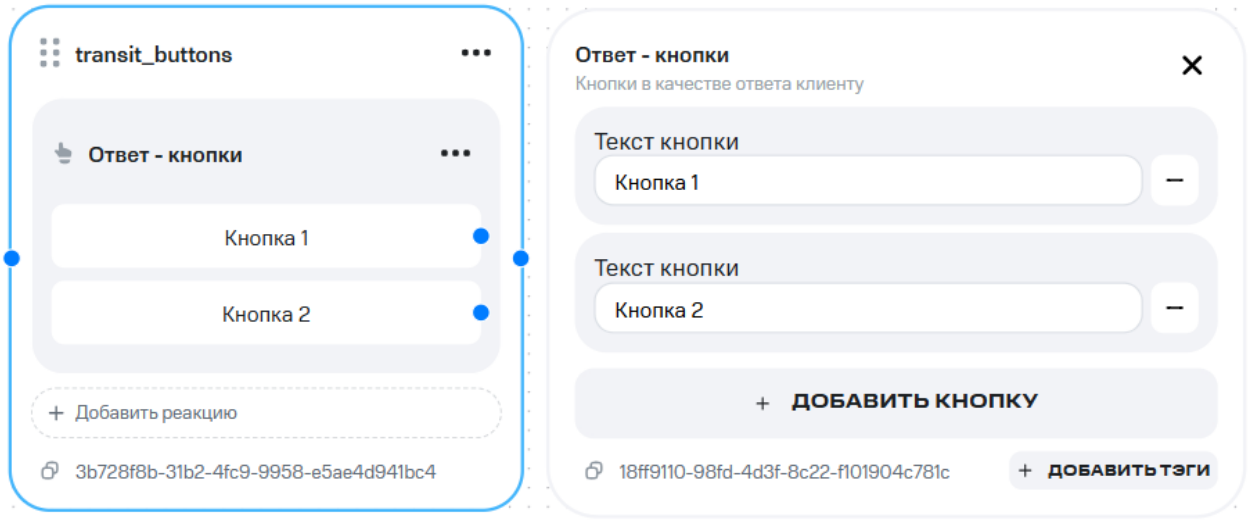
Пример.

«Количество символов в вашем сообщении превышает {{max}}, оно составляет {{system.last_user_message_length}}. Попробуйте перефразировать»

В данном случае **max** и **system.last_user_message_length** – имена зарезервированных переменных или переменных, которые будут сохранены в контекст пользователя другими реакционными блоками в процессе диалога. При выполнении данного реакционного блока вместо названий переменных подставляются их значения.

Кнопки

Кнопки в качестве ответа клиенту.

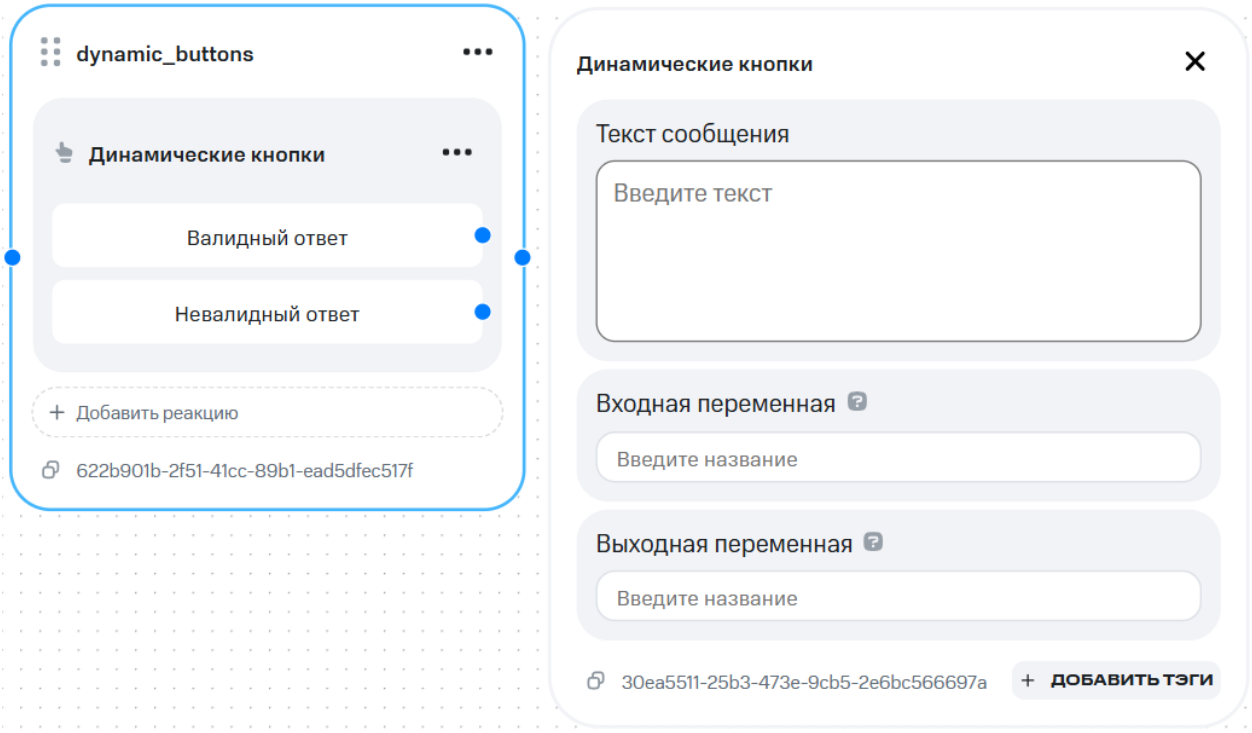


Значения задаются в сценарии. В поле **Текст кнопки** укажите название кнопки, которое нужно отобразить в чате для пользователя. Нажмите **Добавить кнопку**, чтобы добавить необходимое количество кнопок.

Если нужны кнопки, которые формируются автоматически, воспользуйтесь блоком **Динамические кнопки**.

Динамические кнопки

Кнопки, которые автоматически формируются на основе полученных данных в ходе выполнения сценария.



ВНИМАНИЕ

При разработке сценария рекомендуется самостоятельно проверять количество кнопок, иначе их может сформироваться большое количество.

Кликните на блок и заполните поля:

- **Текст сообщения** – текст, который должен отображаться перед списком кнопок;
- **Входная переменная** – переменная, из которой нужно формировать список кнопок для отображения. Обязательное поле для заполнения. Формат значения переменной:

```
{
    "<Кнопка 1>": <Значение>,
    "<Кнопка 2>": <Значение>,
    ...
     "<Кнопка N>": <Значение>
}
```

Где:

Кнопка 1, **Кнопка 2**, **Кнопка N** – названия кнопки; **Значение** – значение, которое сохраняется при выборе кнопки.

Пример заполнения входной переменной:

```
{
    "Клиника 1": 10,
    "Клиника 2": 12,
    "Другое": -1
}
```

- **Выходная переменная** – переменная, в которую сохраняется значение выбранной пользователем кнопки. Например, если нажали на кнопку **Клиника 1**, то в выходную переменную сохраняется значение **10**. Указанную переменную можно использовать в следующих блоках сценария.

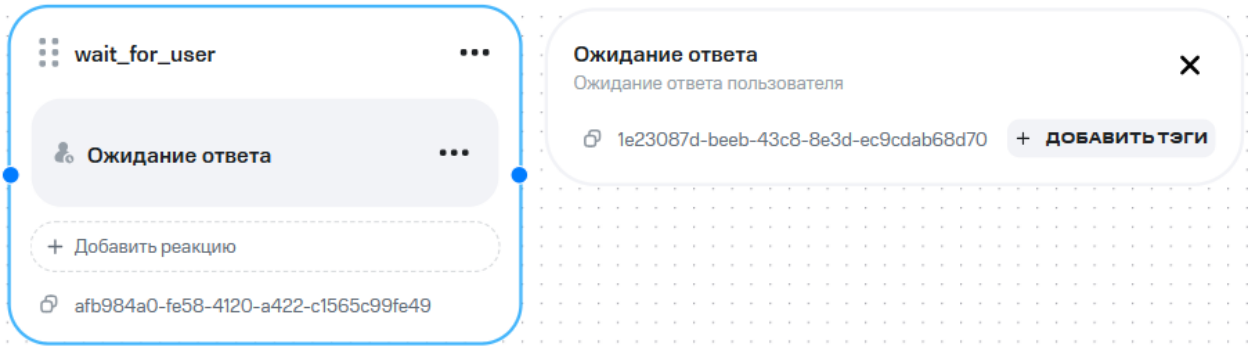
Добавьте связи в зависимости от ответа:

- **Валидный ответ** – связь с реакционными блоками, которые должны выполниться, если пользователь выберет одну из динамических кнопок;
- **Невалидный ответ** – связь с реакционными блоками, которые должны выполниться, если от пользователя придет отличное от названий динамических кнопок значение.

Если получен валидный или невалидный ответ, связь для которого не добавлена, то после блока **Скрипт** выполняется переход к следующему реакционному блоку.

Ожидание ответа

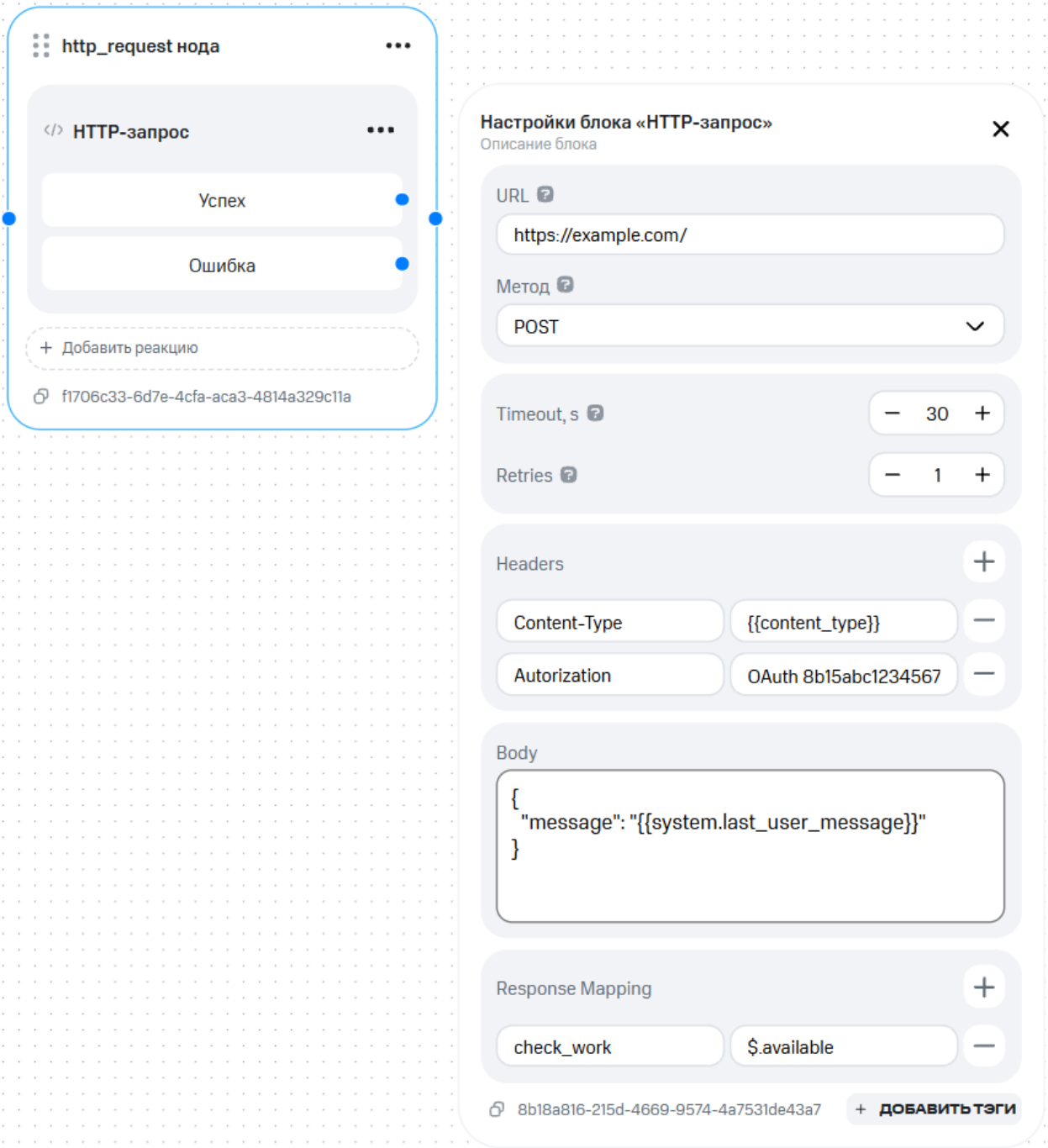
Блок ожидания ответа от пользователя.



Используйте блок, если нужно дождаться ответа, прежде чем переходить к следующим блокам сценария.

HTTP-запрос

Выполнение HTTP-запроса.



В параметрах блока укажите:

- **URL** – конечная точка запроса (эндпоинт);
- **Метод** – метод запроса, например POST. Выберите HTTP-метод из выпадающего списка;
- **Таймаут, секунды** – максимальное время ожидания ответа на запрос в секундах. Если запрос завершается по тайм-ауту, то выполняется переход к стейту обработки невалидного ответа;
- **Retries** – количество попыток повторить неуспешный запрос. По умолчанию **0** – неуспешный запрос не повторяется. Чтобы повторить запрос один раз, укажите 1;
- **Headers** – заголовки запроса. Включает в себя название заголовка (key) и значение (value). По кнопкам + и - вы можете добавить или удалить строки для заполнения;
- **Body** – тело запроса в формате JSON. Пример:

```
{
  "message": "{{system.last_user_message}}"
}
```

Где **last_user_message** – имя зарезервированной переменной или переменной, которая будет сохранена в контекст пользователя другим реакционным блоком в процессе диалога. При выполнении данного реакционного блока вместо имени переменной подставляется ее значение;

- **Response Mapping** – маппинг значений параметров, полученных в ответе на запрос, на переменные движка/контекста пользователя. Включает в себя название переменной (key) и описание пути для извлечения и записи значения (value). Переменные сохраняются в контекст текущего диалога. По кнопкам + и - вы можете добавить или удалить строки для заполнения. В значениях доступны опции:
  - $response.<...> или $response.body.<...> – извлечение значений переменных из body ответа;
  - $response.headers.<...> – извлечение значений переменных из headers ответа;
  - $response.status_code – извлечение HTTP-кода ответа.

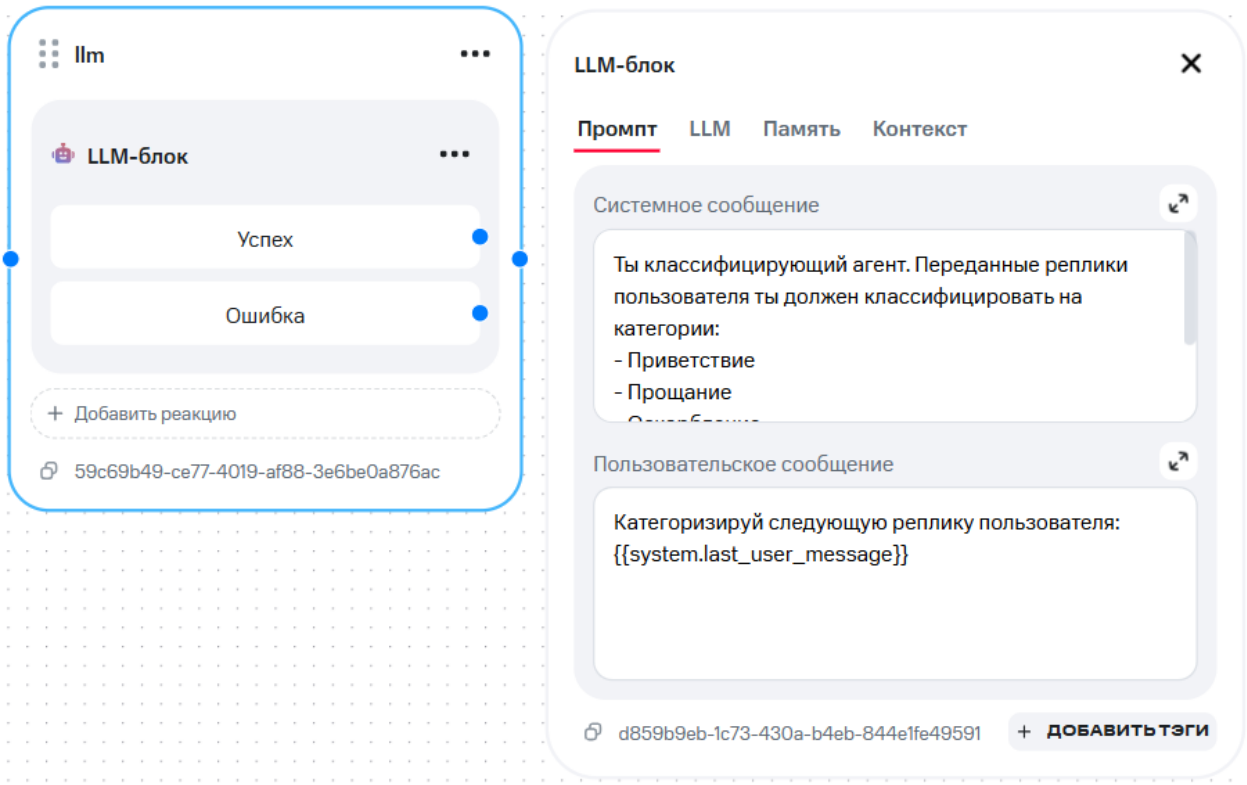
> **ПРИМЕЧАНИЕ**
>
> В полях **URL**,** Header**, **Body** вы можете использовать переменные, заключенные в двойные фигурные скобки: **{{Имя переменной}}**.

Добавьте связи для блока:

- **Успех** – с блоком реакции, к которому нужно перейти, если запрос выполнен успешно;
- **Ошибка** – с блоком реакции, к которому нужно перейти в случае неудачи при выполнении запроса.

LLM

Блок позволяет решать задачи, связанные с пониманием естественного языка и генерации ответа.



Примеры задач: перевод текста, генерация описания какого-либо объекта, ответ на конкретный вопрос клиента, генерация краткого содержания длинного текста. Результат работы большой языковой модели записывается в переменную, которую можно использовать в следующих реакционных блоках.

> **ВНИМАНИЕ**
>
> Для решения более сложных задач с применением дополнительных инструментов используйте блок AI-агент.

Чтобы настроить отправку запроса в LLM:

1. Заполните промпт для модели:
   - **Системное сообщение** – инструкция для модели, в которой определяется роль, правила и цели модели, в том числе возможности и ограничения;
   - **Пользовательское сообщение** – контекст, в котором может содержаться входящее сообщение клиента или другие данные, которые модель должна обработать.

   > **ПРИМЕЧАНИЕ**
   >
   > В промпте вы можете использовать переменные из текущего проекта, заключенные в двойные фигурные скобки: {{Имя переменной}}.

2. На вкладке **LLM** заполните настройки подключаемой модели:

НАЗВАНИЕ ПАРАМЕТРА	ОПИСАНИЕ
URL	Ссылка на модель

НАЗВАНИЕ ПАРАМЕТРА	ОПИСАНИЕ
Ссылка на модель	Токен доступа к модели. Необязательный параметр
Модель	Имя модели, к которой выполняется запрос. Поддерживаются OpenAI-совместимые API-модели
Timeout, s	Максимальное время ожидания ответа от модели за одну попытку. Указывается в секундах
Retries	Количество попыток получить ответ от модели, если при обращении к ней возникает ошибка
Max number of tokens	Максимальное количество токенов, сгенерированное LLM в ответ на запрос
Sampling temperature	Уровень случайности в ответах, генерируемых большой языковой моделью. Высокие температуры приводят к более разнообразным и креативным результатам, в то время как низкие температуры – к более консервативным и предсказуемым реакциям. Возможные значения: от 0.0 до 2.0
Top K	Количество наиболее вероятных токенов, которые модель учитывает при генерации текста. Чем ниже значение **Top K**, тем более предсказуемым и повторяющимся будет ответ модели. Возможные значения: от 1 до 10000
Top P	Пороговая вероятность включения токенов в набор кандидатов, используемый LLM для генерации выходных данных. Под токеном понимается минимальная единица текста, с которой способна работать языковая модель. Низкие значения параметра **Top P** приводят к более точным и основанным на фактах ответам от LLM, тогда как более высокие значения увеличивают случайность и разнообразие в сгенерированном ответе. Возможные значения: от 0.1 до 1.0
Frequency penalty	Штраф за повторение токенов в тексте в зависимости от частоты появления. Токены, которые встречаются в тексте чаще, с меньшей вероятностью будут использоваться ИИ снова. Параметр позволяет уменьшить частоту повторений. Возможные значения: от -2 до 2
Presence penalty	Фиксированный штраф за повторение токенов, независимо от частоты появления.Возможные значения: от -2 до 2
Extra body	Дополнительные параметры, которые нужно дополнительно передать в запросе к LLM-серверу помимо стандартных полей. Задается в формате JSON. Необязательное поле

ПРИМЕЧАНИЕ

Если в ваш комплект поставки входит большая языковая модель Cotype Pro, то на вкладке **LLM** вы можете указать ее. Для этого заполните поля:

- **URL** – введите адрес до модели в формате: <Путь до модели>:8080/v1/, например http://cotype.dev-mars:8000/v1. Адрес по умолчанию можно посмотреть в файле values,yaml сервиса rag-manager в параметре **LLM_URL**;
- **Модель** – имя используемой версии Cotype Pro, например cotype_pro_2_mars.

1. На вкладке **Память** настройте контекст для модели. Установите ползунок **Итерация взаимодействия** на том значении, которое обозначает количество последних итераций взаимодействия клиента и бота. Возможные значения: от 0 до 100.
2. На вкладке **Результат** в поле **Переменная** укажите имя переменной. В эту переменную сохраняется результат работы LLM. Ее можно использовать в следующих блоках сценария. Например, добавьте блок **Текст** для отображения результата работы LLM в чате с пользователем.

   При необходимости установите флажок **Стриминг ответа на поверхность**. В этом случае ответ от модели будет возвращаться на поверхность частями (чанками) в режиме реального времени.

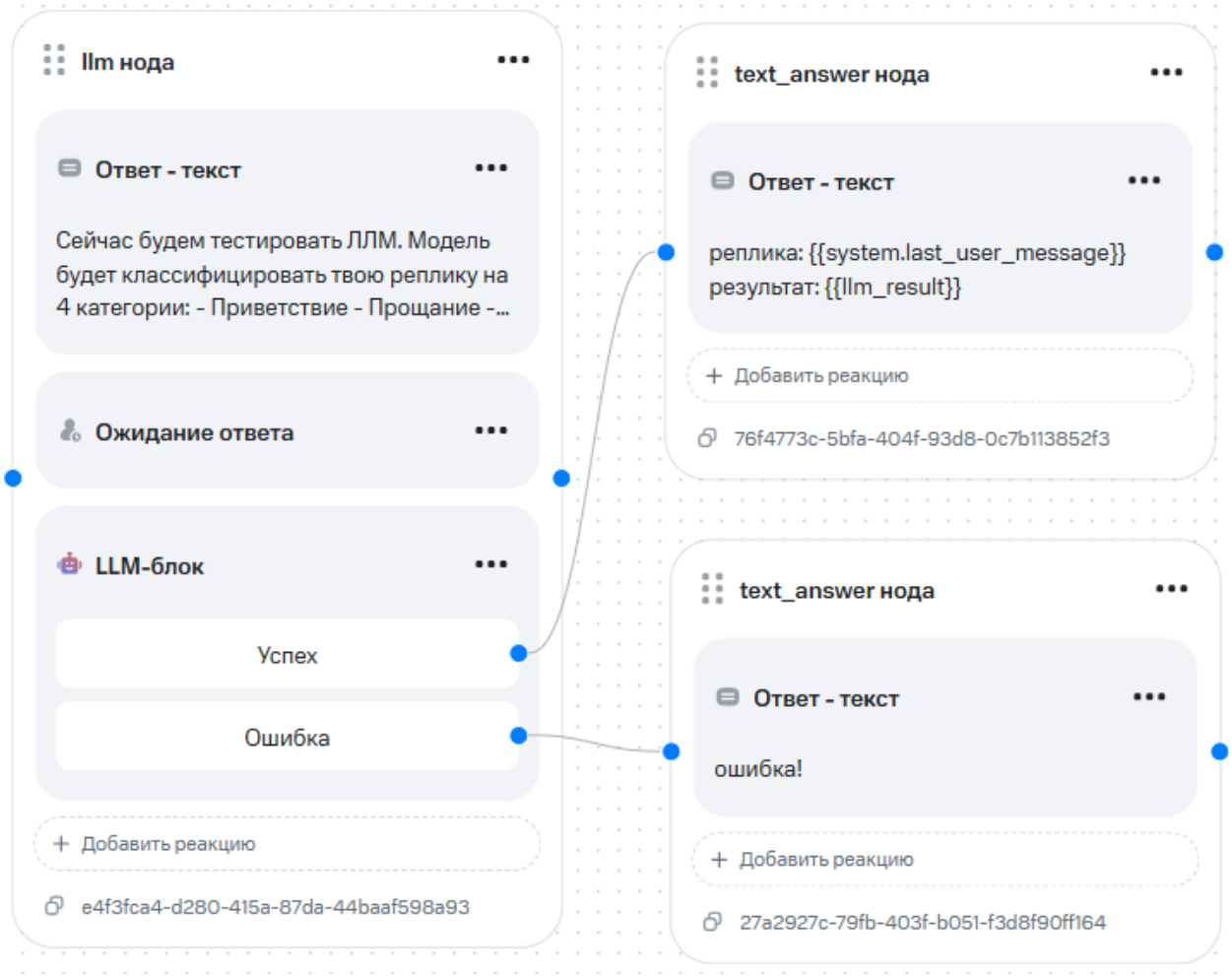
   ВНИМАНИЕ

   Стриминговый режим поддерживается для каналов с типами HTTP, Telegram и Max. В тестовом виджете ответ присылается только после полного его формирования.

   Если у вас включен стриминг ответа, то добавлять текстовый блок для вывода ответа в чате не нужно. Ответ от модели отправится в чат автоматически. Если текстовый блок для вывода добавлен, то ответ продублируется.

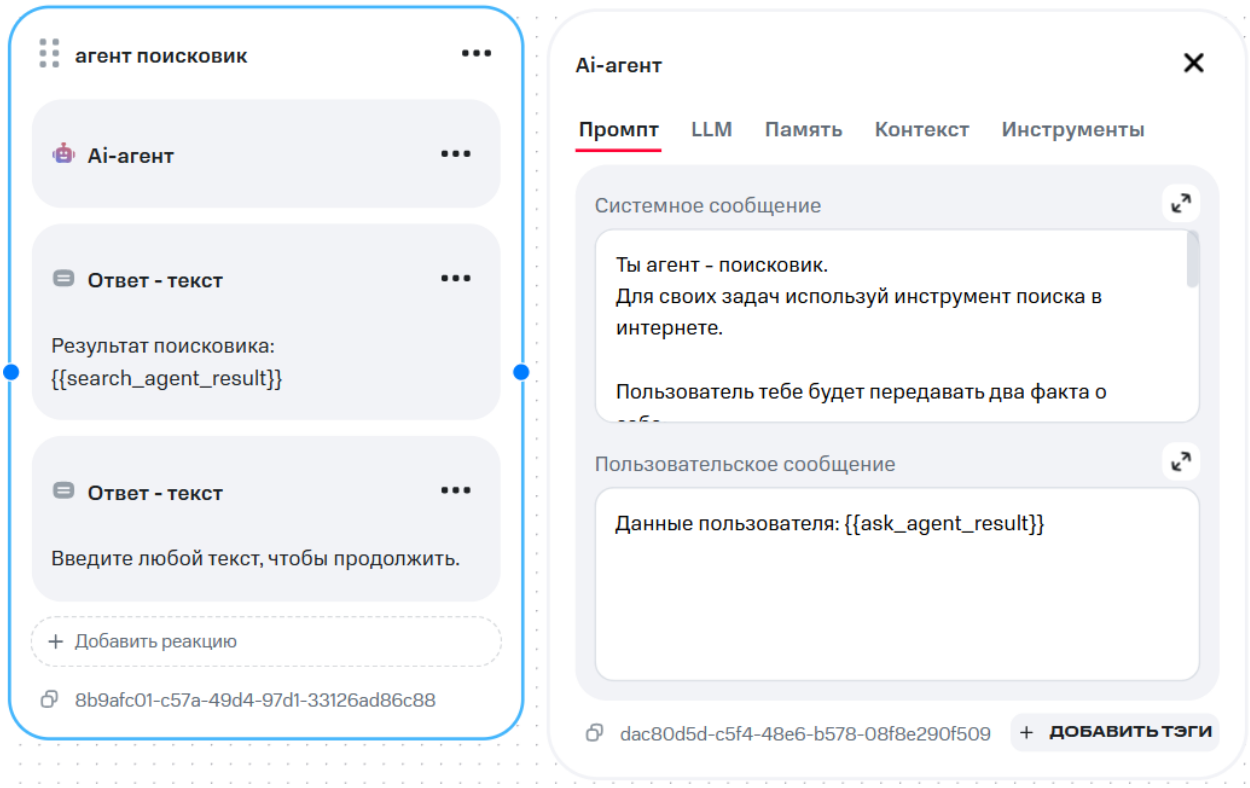
3. Добавьте связи для блока:
   - **Успех** – переход в реакционный блок, если блок LLM выполнен успешно;
   - **Ошибка** – переход в реакционный блок, если во время выполнения блока LLM возникла ошибка.

Пример создания блока LLM:



AI-агент

Блок для отправки запроса в большую языковую модель (LLM) и генерации ответа с подключением дополнительных инструментов.



Инструменты — это вспомогательные ресурсы, которые позволяют агенту решать задачи за пределами возможностей LLM. Например, искать информацию в интернете, выполнять математические вычисления, сохранять информацию в базе данных и т.д.

Если проект с блоком **AI-агент** подключен к каналу с типом HTTP, то сгенерированные агентом уточняющие вопросы в чате с пользователем приходят в стриминговом режиме. Это значит, что текст отправляется чанками, и задержка времени на формирование полного ответа не происходит. Для других типов каналов и в тестовом виджете стриминговый режим не поддерживается.

> **ВНИМАНИЕ**
>
> Если предполагается только работа с естественным языком, то вместо блока AI-агент используйте блок LLM.

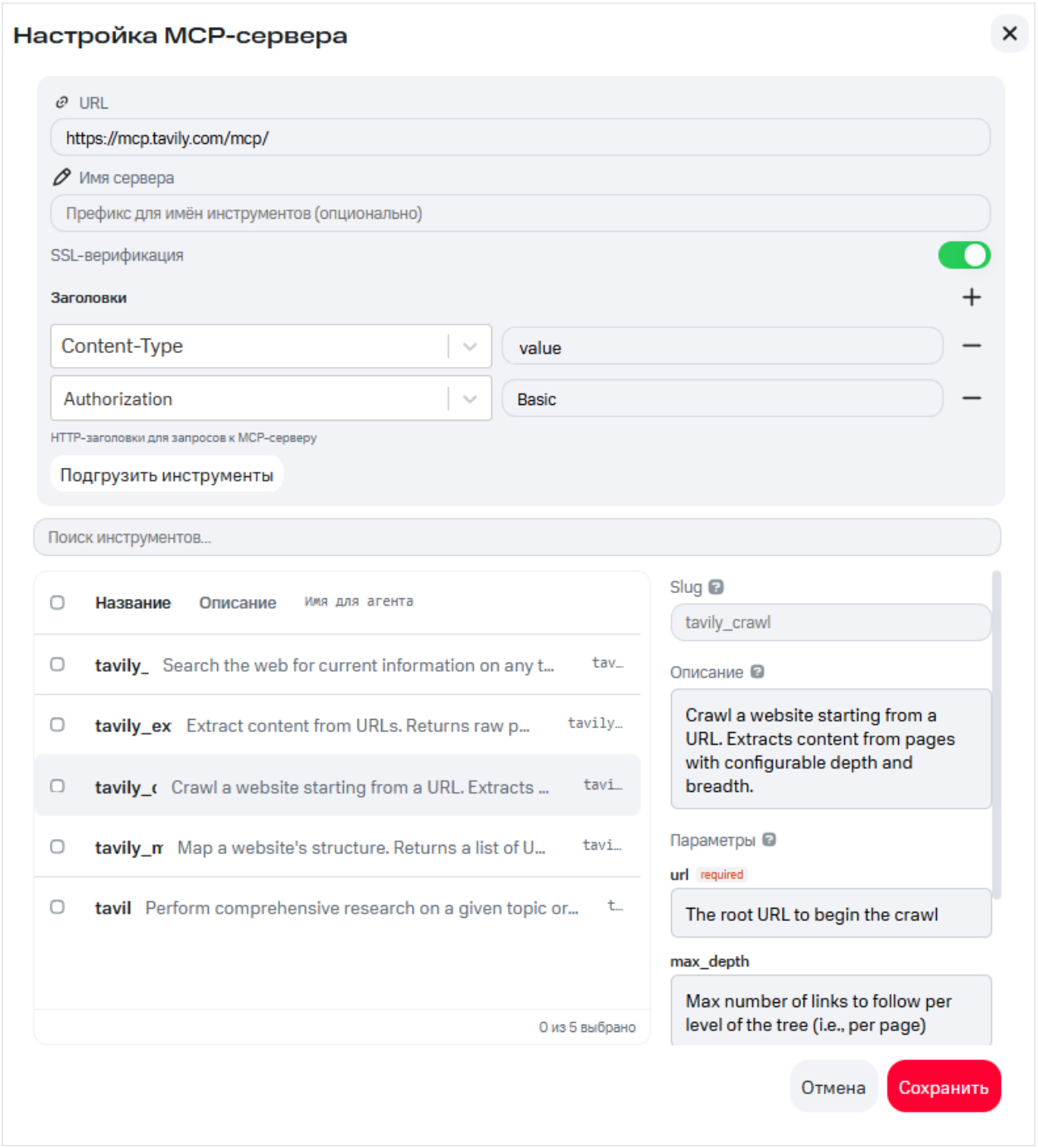
1. Заполните поля на вкладках **Промпт**, **LLM**, **Память**, **Контекст**. Подробнее см. в описании блока LLM.
2. На вкладке **Инструменты** выберите инструмент:
   - MCP-сервер;
   - Custom tool;
   - Need clarification.
3. В зависимости от инструмента нажмите на кнопку ⌄ или ⚙ заполните поля.
4. Ограничьте количество повторных вызовов одних и тех же инструментов в рамках одного обращения к блоку, если необходимо. Настройки позволяют контролировать количество обращений агента к инструменту и предусматривать поведение при превышении лимита.

ПОДСКАЗКА

Если проект подключен к каналу с типом HTTP, Telegram или MAX, то сгенерированные агентом ответы и уточняющие вопросы в чате с пользователем приходят в стриминговом режиме, даже если не включена соответствующая настройка на вкладке **Результат**. Это значит, что текст отправляется чанками, и задержка времени на формирование полного ответа не происходит. Для других типов каналов и в тестовом виджете стриминговый режим не поддерживается.

MCP-сервер

При выборе данного типа инструмента подключение к нему выполняется по URL.

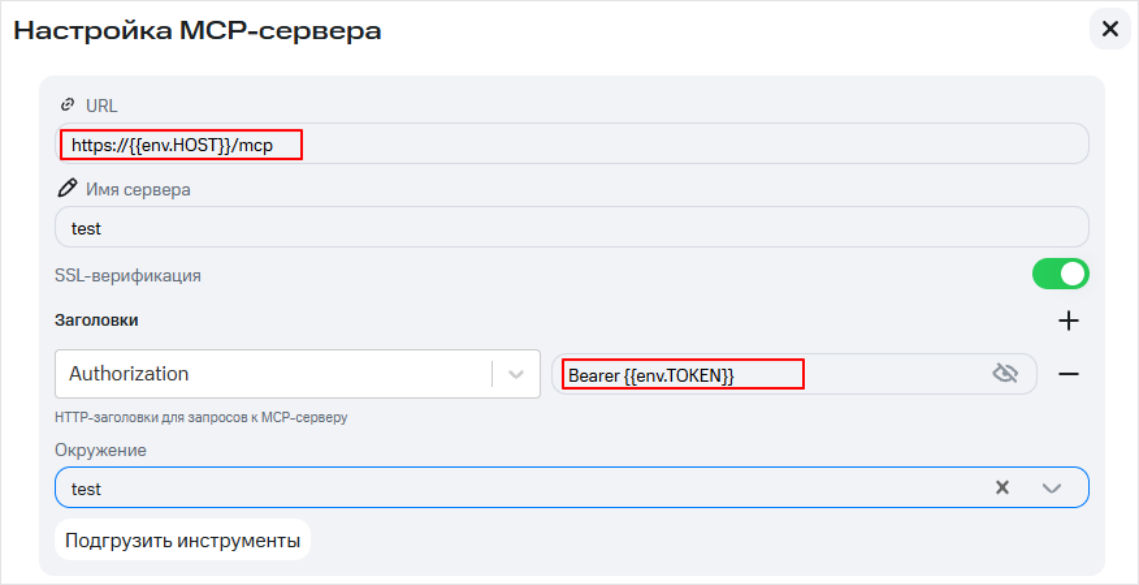


1. Заполните поля:
   - **URL** – адрес MCP-сервера, который нужно использовать при обработке запроса
   - **Имя сервера** – имя, которое нужно использовать в качестве префикса для инструментов этого сервера. Это позволит модели сориентироваться в случае одинаковых названий инструментов. Например, для первого MCP-сервера добавьте префикс «MCP-1», а для второго – «MCP-2»;
   - **SSL-верификация** – измените положение переключателя в зависимости от того, нужно ли использовать SSL-верификацию;
   - **Заголовки** – пары ключей и значений в формате текстовых строк, передаваемые в HTTP-запросе.

   Нажмите на кнопку +, чтобы добавить строку, — – если строку нужно удалить.

ПОДСКАЗКА

В полях **URL** и **Заголовки** можно использовать переменные окружения. При этом имена переменных указывайте через обращение к скоупу env. Пример заполнения:



2. Нажмите на кнопку **Подгрузить инструменты**, чтобы проверить подключение. В результате отображается таблица с загруженными инструментами.
3. В списке выберите нужные инструменты для более качественной работы агента. Отметьте их флажками.

   При необходимости вы можете изменить детальные настройки инструмента:

   - **Slug** – название инструмента. Измените значение параметре, если существующее не подходит;
   - **Описание** – описание инструмента;
   - **Параметры** – описания параметров. Вы можете скорректировать описания для корректного заполнения параметров моделью.
4. Нажмите на кнопку **Сохранить**.

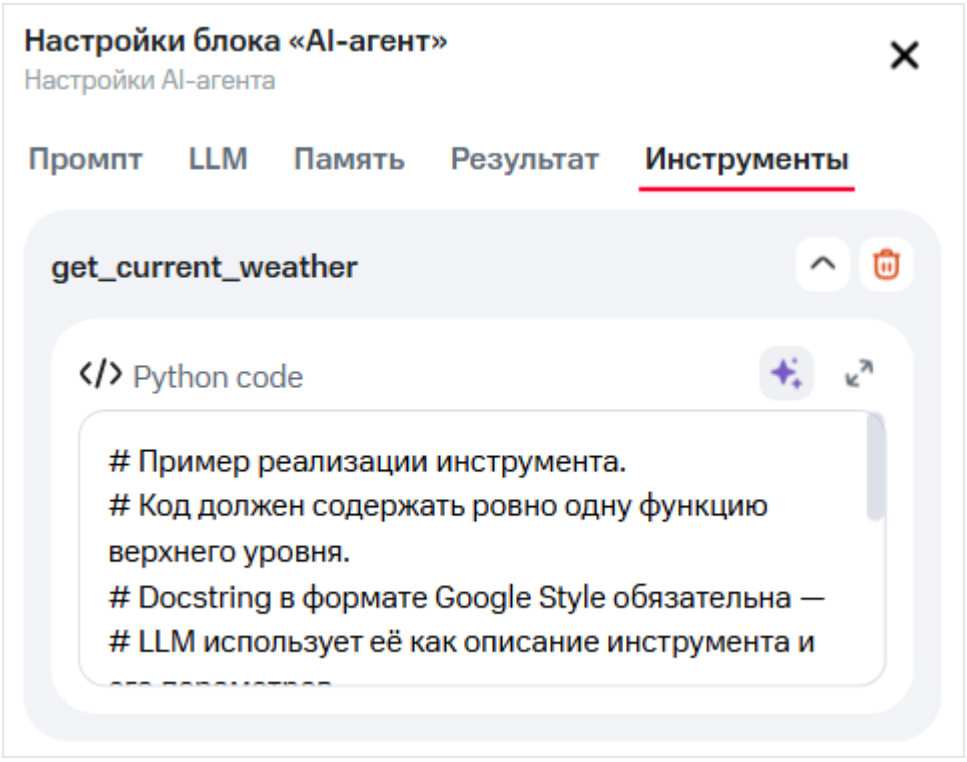
ВНИМАНИЕ

Поддерживается только транспортный протокол Streamable HTTP.

Custom tool

Инструмент позволяет вызвать внешний сервис из программного кода. В нем вы можете указать вызов API и порядок обработки запроса, стандартизировать общение агентов с инструментами, а также прописать обработку ошибок при сбоях.

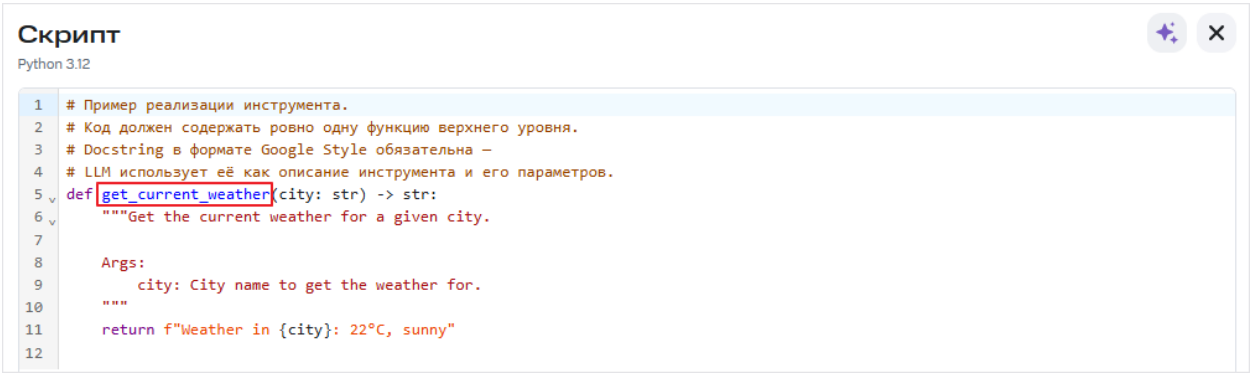
По умолчанию в поле **Python code** приведен пример кода для инструмента:



Для удобства можно развернуть окно ввода кода на весь экран по кнопке .

В коде должна содержаться ровно одна функция верхнего уровня. Также нужно указать строковую переменную docstring в формате Google Style, значение которой заключается в тройные кавычки. Она будет использоваться в качестве описания инструмента и его параметров.

Чтобы изменить имя инструмента, измените имя функции в коде:



ПОДСКАЗКА

В программном коде вы можете использовать функции следующих модулей: array, base64, binascii, bisect, calendar, cmath, collections, contextlib, contextvars, copy, csv, dataclasses, decimal, email, encodings, enum, fractions, functools, hashlib, heapq, hmac, html, ipaddress, itertools, json, keyword, math, numbers, pprint, quopri, re, reprlib, secrets, shlex, statistics, string, textwrap, tomllib, typing, unicodedata, uuid, zoneinfo, datetime. Они уже подключены, и дополнительно использовать оператор import не нужно.

Подробнее о функциях модулей см. статью The Python Standard Library в официальной документации Python.

Если к агенту подключено несколько инструментов, то их имена могут быть одинаковыми.

Пример для тестирования получения погоды

```
async def fetch_weather_by_city_code(city_code: str):
    """
    Моковый инструмент для получения погоды по коду города

    Args:
        city_code: Код города
    """

    async with aiohttp.ClientSession() as session:
        async with session.get(
            f"https://postman-echo.com/get",
            params={"value": city_code},
        ) as response:
            data = await response.json()

    return {
        "passed_args": data["args"],
    }
```

В коде инструмента можно обращаться к переменным контекста. Например, получить имя пользователя из сессии или сохранить результат вычислений в переменную, которая будет доступна в следующих блоках сценария. Для этого в функцию нужно добавить параметр с типом Context. Если параметр не указан, то инструмент не получит доступ к контексту.

> **ПОДСКАЗКА**
>
> Имя параметра может быть любым. Он может находиться в любой позиции среди аргументов. При этом языковая модель не видит этот параметр, он не попадает в описание инструмента и не заполняется моделью.

Системный промпт агента обновляется на каждой итерации цикла. Это значит, что если во время работы инструмента изменилось значение какой-либо переменной контекста, которая используется в промпте, то языковая модель на следующем шаге получит обновленное значение.

Если одновременно используется несколько инструментов и каждый из них изменяет контекст, то применяются изменения последнего завершившего инструмента.

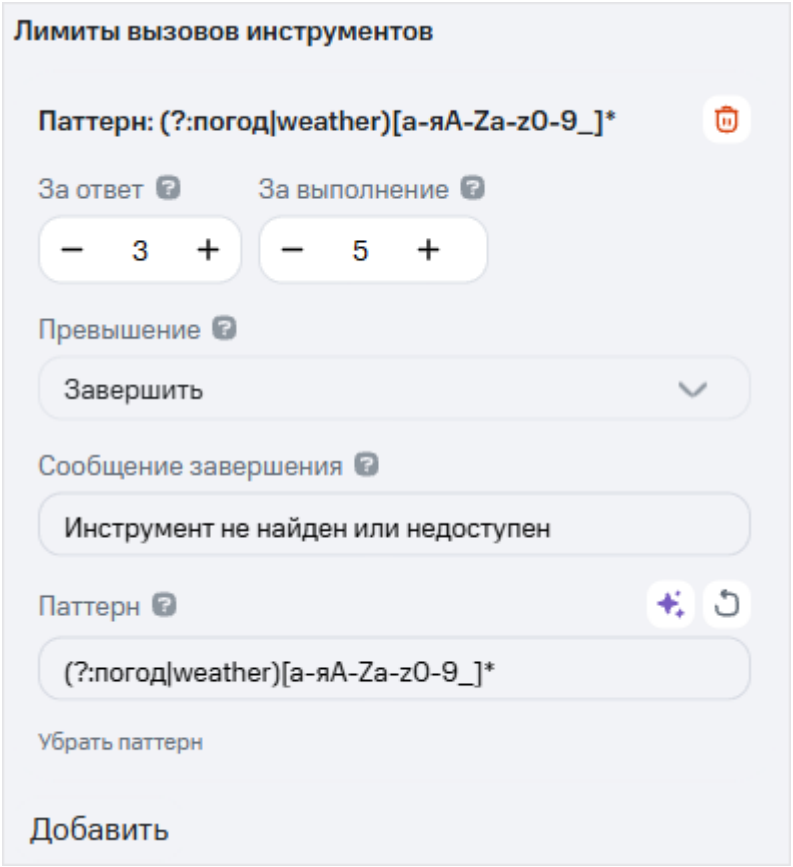
Пример использования переменной контекста

```
def get_weather(city: str, ctx: Context) -> str:
    """Get weather for a city."""
    user_name = ctx.session.user_name
    ctx.session.last_weather_city = city

    return f"Weather in {city} for {user_name}: sunny"
```

Настройка лимита вызовов

Чтобы настроить лимиты, в поле **Лимиты вызовов инструментов** нажмите **Добавить** и заполните поля:



- **За ответ**. Максимальное число вызовов за одну генерацию ответа агента. Счётчик сбрасывается, когда срабатывает инструмент need_clarification и агент начинает новую генерацию. По умолчанию — 5;
- **За выполнение**. Максимальное число вызовов за всё время работы блока, включая все циклы с need_clarification. Счётчик не сбрасывается до завершения блока;
- **Превышение**. В выпадающем списке выберите поведение агента при превышении лимита:

  Продолжить (по умолчанию) — агент получает сообщение, что лимит исчерпан, и продолжает работу: перестает вызывать заблокированный инструмент, может использовать другие инструменты или дать ответ самостоятельно;

  Ошибка — агент немедленно завершает работу с ошибкой;

  Завершить — агент останавливается без ошибок и возвращает пользователю текст, заданный в поле **Сообщение завершения**.
- **Сообщение завершения**. Текст, который будет отображаться пользователю при остановке агента. Поле отображается, если при превышении лимита выбрано поведение **Завершить**.
- **Паттерн**. Регулярное выражение, по которому фильтруются имена инструментов. Настройки лимита будут применяться к вычисленным инструментам. Если паттерн не указан, то лимит действует для всех инструментов.

ПОДСКАЗКА

Для составления регулярного выражения в поле **Паттерн** вы можете использовать AI-ассистент.

В результате при каждом вызове инструмента проверяются все заданные лимиты. Если инструмент подходит под указанное регулярное выражение или паттерн не задан, а также один из счетчиков **За ответ** или **За выполнение** достиг предела, то срабатывает выбранное поведение.

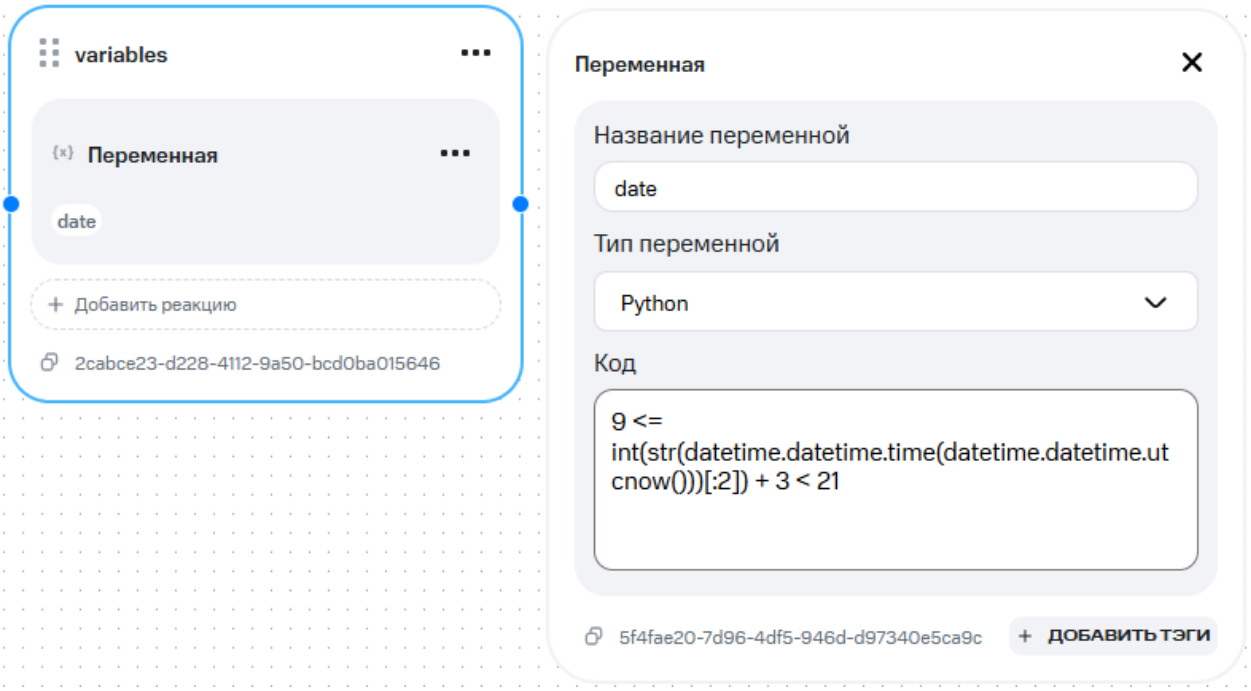
Need clarification

Если инструмент включен, то во время работы агент запрашивает у пользователя недостающую информацию.

По умолчанию промпт для этого инструмента уже задан, вы можете его изменить в поле **Описание инструмента**. Также в соответствующем поле вы можете изменить описание параметра query.

Переменная

Создание и сохранение переменных в диалог пользователя.



Чтобы добавить переменную, нажмите на блок **Переменная** и в открывшемся окне заполните поля:

- **Название переменной**;

ВНИМАНИЕ

Если переменная хранится не в области session, то в ее названии указывайте скоуп, например temp.value. Подробнее см. в разделе «Области видимости и времени жизни переменных».

- **Тип переменной**. Выберите значение из списка. Доступные типы:
  - **Constant** – константа. Значение по умолчанию. В качестве значения переменной используется то, что указано в поле Код;
  - **Python** – код на языке Python. В этом случае значение поля **Код** будет отправляться на вычисление в интерпретатор кода;
  - **Regexp** – регулярное выражение. В качестве значения используется результат вычисления регулярного выражения;
  - **Regexp map** – массив регулярных выражений. В качестве значения переменной используется результат первого сработавшего регулярного выражения.

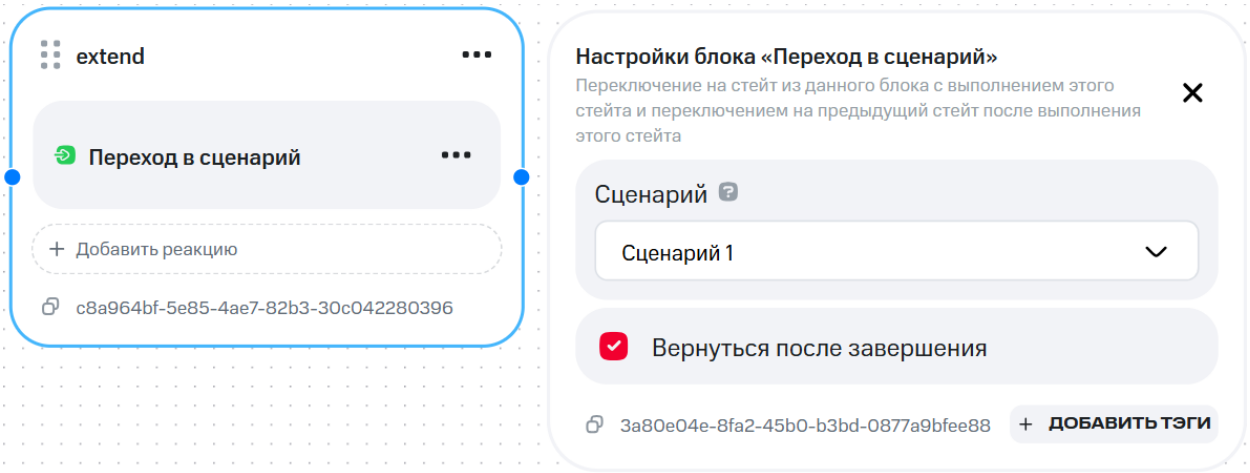
> **ВНИМАНИЕ**
>
> Если тип переменной **Regexp** или **Regexp map** и после выполнения регулярных выражений значение переменной не определено, то в качестве него используется **null**. Это происходит, если запрос пользователя не подходит ни под одно регулярное выражение.

- **Код** – значение переменной. Может быть задано константой, регулярным выражением или кодом на языке Python. Пример значения:

```
9 <= int(str(datetime.datetime.time(datetime.datetime.utcnow()))[:2]) + 3 < 21
```

Переход в сценарий

Вызов другого сценария из данного блока.



В выпадающем списке **Сценарий** отображается список всех сценариев бота. Выберите тот, к которому нужно перейти после выполнения текущего блока. Если по завершению выбранного сценария нужно выполнить переход к исходному, установите флажок **Вернуться после завершения**. В этом случае сценарий продолжится со следующего блока.

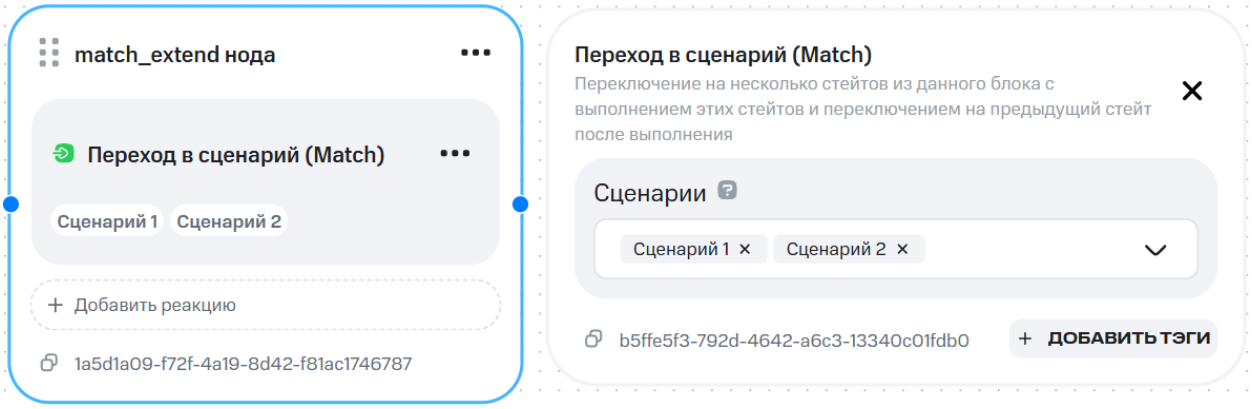
Если нужно перейти к конкретной ноде другого сценария, то используйте метод контекста **context.defer_jump_to** в блоке **Скрипт**.

ВНИМАНИЕ

Если блок **Переход** в сценарий добавлен в сценарий препроцессинга и в нем установлен флажок **Вернуться после завершения**, то сценарий, к которому нужно перейти, также выполняется в рамках препроцессинга. Проверьте, что в нем используются только допустимые блоки: **HTTP-запрос**, **Переменная**, **Переход в сценарий**, **Переход в сценарий (Match)**, **Условие**, **Скрипт**. При попытке выполнить недопустимые блоки возникает ошибка.

Переход в сценарий (Match)

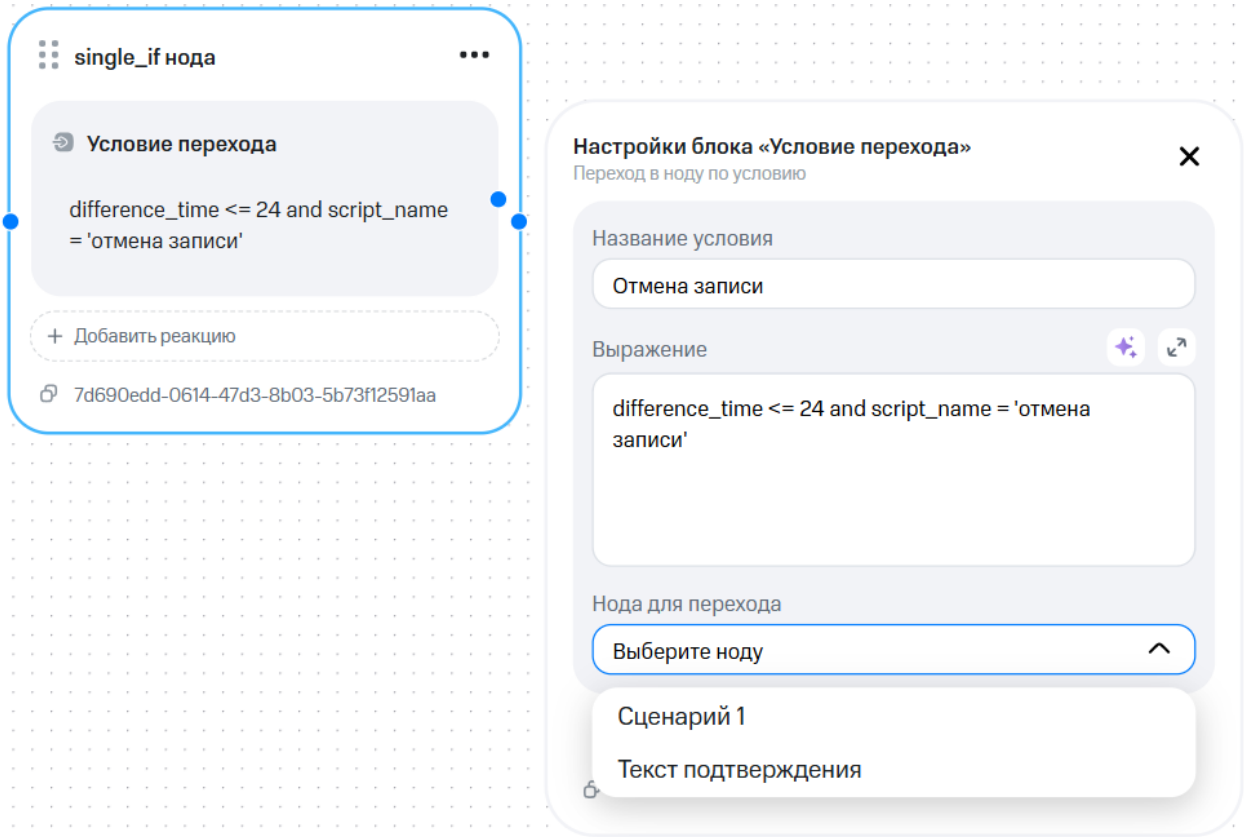
Вызов выбранного сценария из данного блока.



Для выбора указывается несколько сценариев. По их активационным блокам движок определяет, в какой сценарий нужно перейти после выполнения текущего блока. По завершению выбранного сценария выполнится переход к исходному сценарию, он продолжится со следующего блока.

Условие перехода

Условие выбора ноды для перехода.



Заполните поля:

- **Название условия**;
- **Выражение**. В поле укажите условие, которое нужно проверить. При заполнении можно использовать переменные. Например, если нужно проверять последнее сообщение пользователя, укажите переменную **last_user_message** – запрос пользователя или текст кнопки;
- **Нода для перехода**. В выпадающем списке выберите ноду, к которой нужно перейти, если условие истинно.

При заполнении выражения используйте:

- операторы:

  = – сравнение на равенство;

  != – сравнение на неравенство;

  ( ) – выделение конструкций;

  < – меньше;

  <= – меньше либо равно;

  == – сравнение на равенство;

  > – больше;

  >= – больше либо равно;

  + – сложение;

  - – вычитание;

  ***** – умножение;

  / – деление;

  is – проверка совместимости результата выражения с указанным типом;

and – логическое «И»;

or – логическое «ИЛИ»;

not – логическое «НЕ»;

contains – проверка на наличие искомой подстроки в исходной строке;

not_contains – проверка на отсутствие искомой подстроки в исходной строке;

matches – проверка на наличие строки, представленной в виде регулярного выражения.

Пример 1. difference_time <= 24 and script_name = 'отмена записи'

Пример 2. system.united_user_messages matches 'парков(ка|ки|ок)|стоян(ка|ки|ок)|парковочн(ое|ые)'

Пример 3. (system.last_user_message contains 'Купить') or (system.last_user_message contains '1') or (system.last_user_message contains 'дай')

- константы:

null – значение переменной до инициализации;

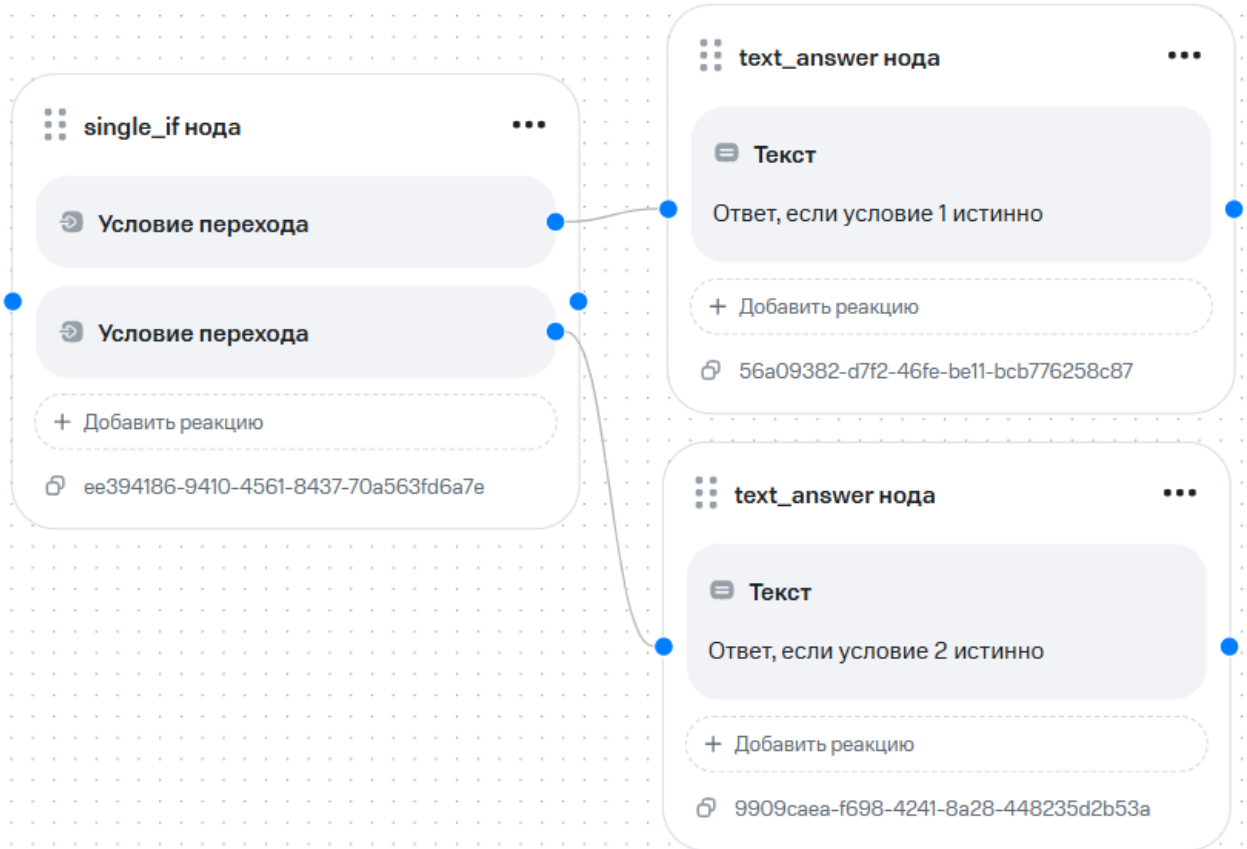
true – «ИСТИНА»;

false – «ЛОЖЬ»;

undefined – отсутствие значения

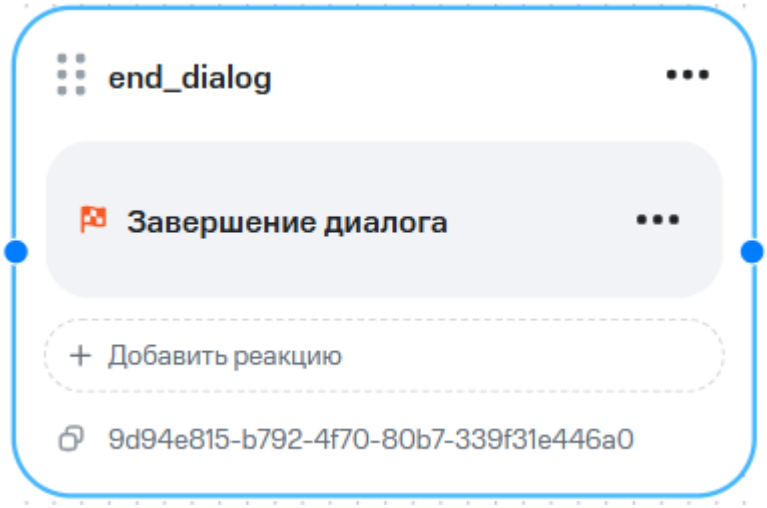
- одинарные кавычки – для указания значений строк. Строка может содержать переменные, заключенные в {{}}. Например, "system.last_user_message = '{{another_var}}'";
- переменные, элементы массива, функцию map. Пример: sarr[1] == "bar" oarr[666].field1 == "foo"

Для обработки ситуации, когда условие ложно, в качестве следующего блока добавьте дополнительный блок **Условие**:

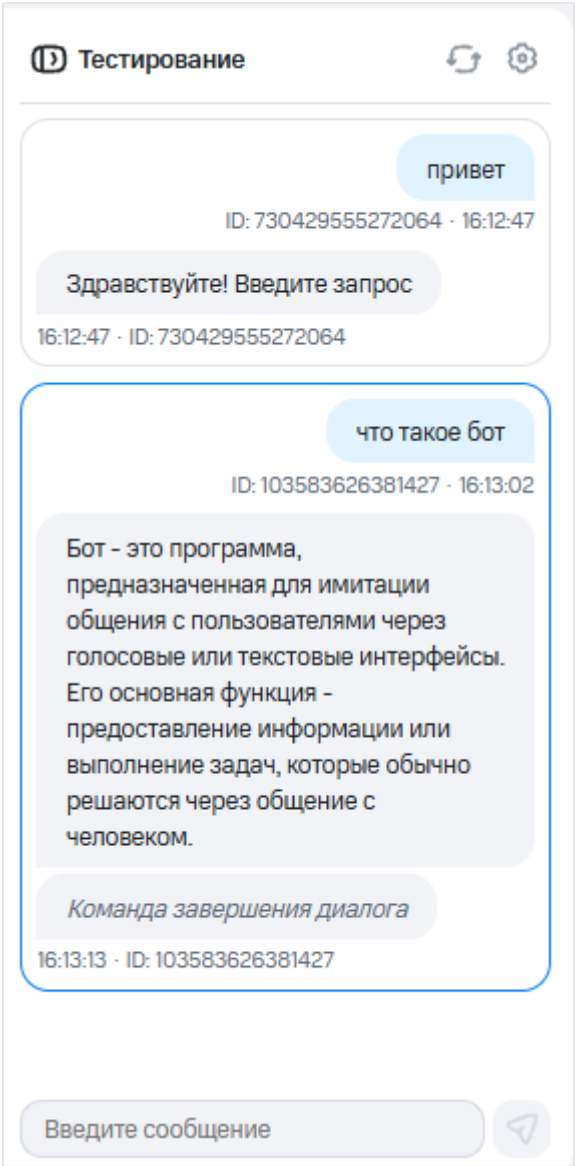


Завершение диалога

Завершение сценария. Добавьте блок и связь с ним, чтобы закончить диалог.

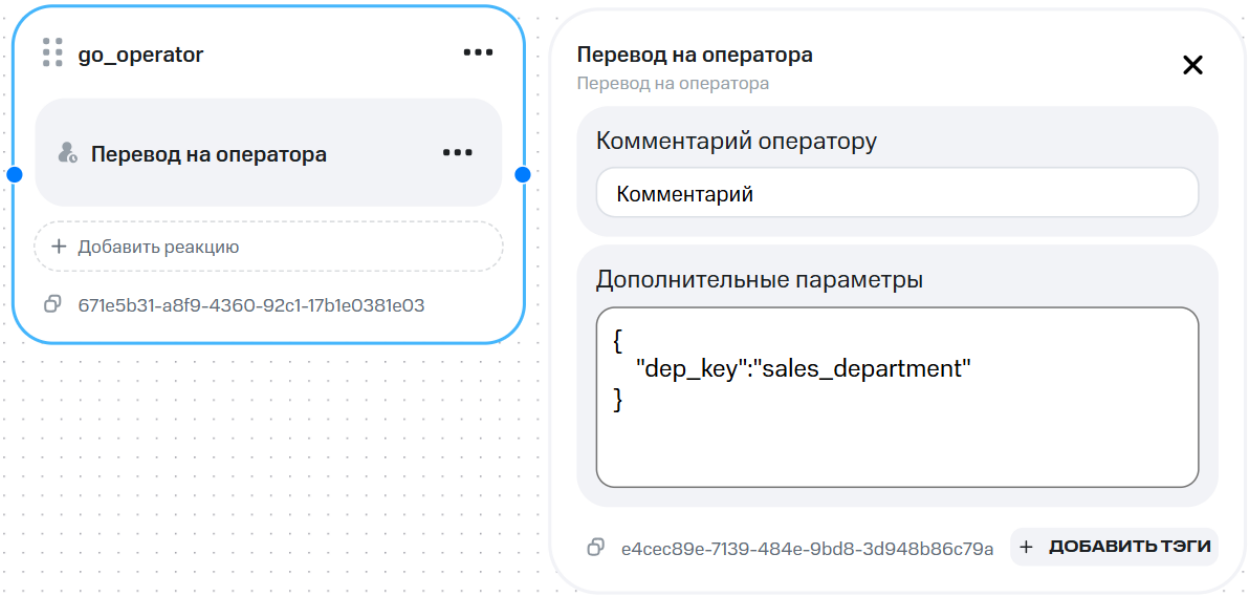


В тестовом виджете для этого блока отображается сообщение: «Команда завершения диалога»:



Перевод на оператора

Перевод диалога на оператора.



Перевод диалога на оператора.

В поле **Дополнительные параметры** в формате JSON можно указать параметры:

- **operator_id** – идентификатор оператора, на которого нужно перевести диалог;
- **dep_key** – идентификатор отдела, в который нужно перевести диалог.

Если указаны оба параметра, то диалог переводится на оператора. В случае ошибки выполняется перевод на отдел. Если снова возникает ошибка, то диалог переводится в общую очередь.

Если параметры не указаны, то также осуществляется перевод в общую очередь.

Пример значения:

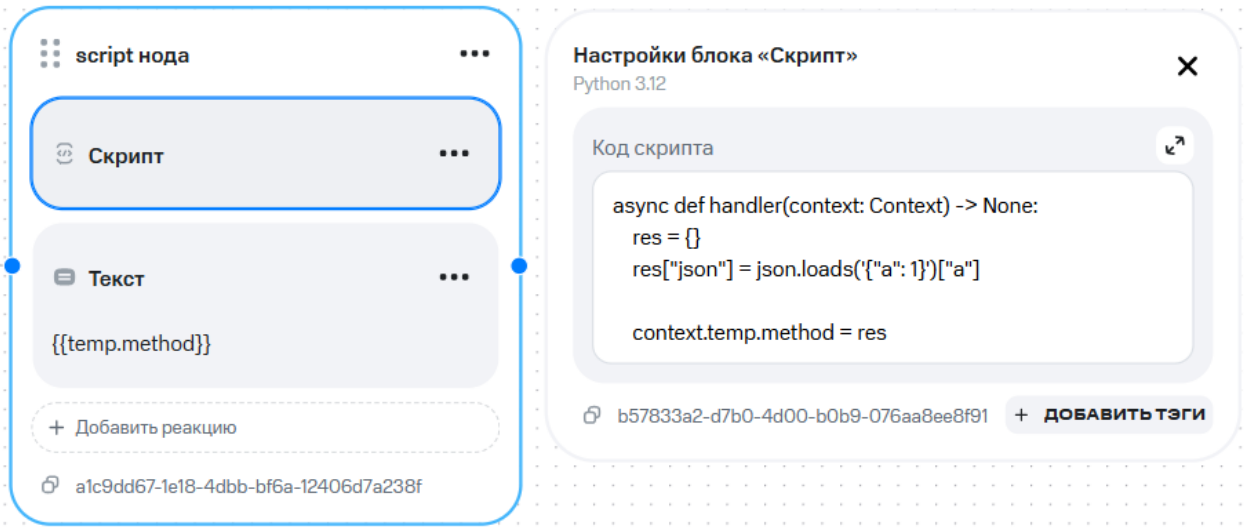
```
{
    "dep_key":"sales_department"
}
```

В поле **Комментарий оператору** можно задать сообщение для оператора.

После блока **Перевод на оператора** сценарий завершается. Блоки, следующие за текущим, не выполняются.

Скрипт

Выполнение произвольного скрипта на языке Python.



В скрипте должна быть определена асинхронная функция handler(context: Context), которая принимает контекст типа dict. В контексте содержатся переменные пользователя, а также доступны зарезервированные переменные движка и вызов предопределенных функций. Контекст, переданный в функцию, можно изменить. В результате заданные переменные становятся доступными в следующих блоках сценария.

> **ВНИМАНИЕ**
>
> При обращении к переменным указывайте области видимости в формате: context.<scope>.<Имя переменной>
>
> Пример: context.system.last_user_message

Для удобства в код скрипта добавлен шаблон, который содержит обязательную функцию обработчика и вспомогательные функции. Заполните код обязательной функции, при необходимости скорректируйте или удалите вспомогательные.

> **ПОДСКАЗКА**
>
> В программном коде вы можете использовать функции следующих модулей: array, base64, binascii, bisect, calendar, cmath, collections, contextlib, contextvars, copy, csv, dataclasses, decimal, email, encodings, enum, fractions, functools, hashlib, heapq, hmac, html, ipaddress, itertools, json, keyword, math, numbers, pprint, quopri, re, reprlib, secrets, shlex, statistics, string, textwrap, tomllib, typing, unicodedata, uuid, zoneinfo, datetime. Они уже подключены, и дополнительно использовать оператор import не нужно.
>
> Подробнее о функциях модулей см. статью The Python Standard Library в официальной документации Python.

Пример

В блоке **Скрипт** задан код:

```
async def handler(context: Context) -> None:
    res = {}
    res["json"] = json.loads('{"a": 1}')["a"]
```

```
    context.temp.method = res
```

В результате выполнения кода создается переменная temp.method, которая становится доступна в следующем реакционном блоке, например в блоке **Текст**.

Предопределенные функции

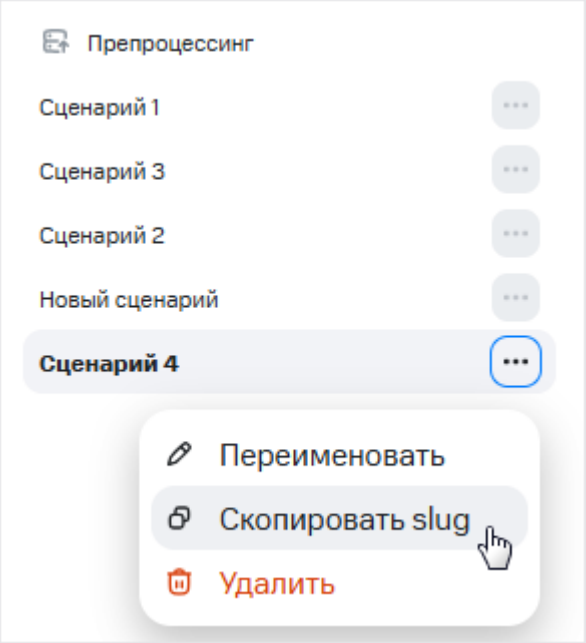
В блоке **Скрипт** вы можете использовать предопределенные функции:

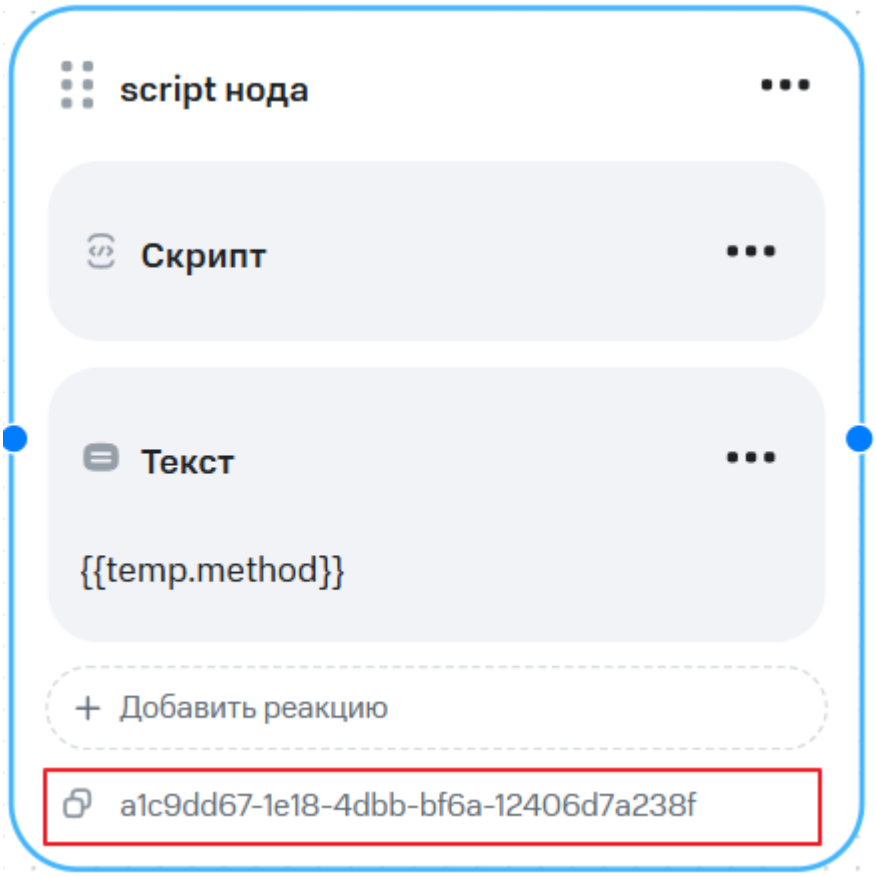
- **defer_jump_to** – перейти к сценарию или ноде;
- **predict_intent** – получить наиболее вероятные интенты для запроса.

defer_jump_to – перейти к сценарию или ноде

Из блока **Скрипт** можно выполнить переход к другому сценарию. Для этого:

1. Скопируйте идентификатор сценария или ноды, к которым нужно перейти. ИД сценария:





ИД ноды:

2. В сценарии, из которого нужно перейти, в блоке **Скрипт** добавьте функцию defer_jump_to:

```
def handler(context: Context) -> None:
 context.defer_jump_to(
     scenario_slug="<scenario_slug>",
     node_id="<node_id>",
     return_back=<True/false>,
 )
```

Где:

scenario_slug – ранее скопированный slug сценария, к которому нужно перейти после выполнения скрипта. Если нужно перейти к конкретной ноде, то переменную не указывайте.

> **ВНИМАНИЕ**
>
> Чтобы обратиться к сценарию, используйте его slug вместо идентификатора, так как слаг остается постоянным, а ИД может измениться, например при импорте сценария.

node_id – идентификатор ноды, к которой нужно перейти. Если нужно выполнить переход по slug, то переменную не указывайте;

return_back – признак того, что после выполнения сценария нужно вернуться в исходную ноду. По умолчанию **False** – возвращаться в исходную ноду не нужно. Если выполняется переход без возврата из сценария препроцессинга, то препроцессинг прерывается.

> **ВНИМАНИЕ**
>
> Если функция defer_jump_to описана несколько раз, то в результате переход будет выполнен к сценарию или ноде, указанным в последней функции.

predict_intent – получить кандидатов для активации по интенту

Функция работает с классификатором, который предвартительно подключен к проекту.

Формат функции:

```
context.nlu.predict_intent(message, top_n=1)
```

Где:

message – сообщение, для которого нужно определить наиболее подходящих кандидатов;

top_n – количество наиболее вероятных кандидатов. Например, в ситуациях, когда одному интенту соответствует сразу несколько блоков активации в разных сценариях, можно указать top_n=1. Это обеспечит наличие только одного подходящего кандидата в списке наиболее вероятных.

Полученные интенты сохраняются в виде массива в переменную **context.nlu.intents**. Элементы массива упорядочены по вероятности и по приоритету иерархии. В переменной **context.nlu.raw_intents** формируется массив из top_n интентов, состоящий из имен интентов и коэффициентов вероятности (score). Для обратной совместимости имя первого интента из массива **context.nlu.raw_intents** и его score сохраняются в переменные **context.system.sure_topic** и **context.system.topic_score** соответственно.

Подробнее об использовании функции **predict_intent** см. в описании блока Интент, раздел «Блоки активации».

Утилитарные функции

Конструктор сценариев поддерживает возможность добавлять утилитарные функции – вспомогательные фрагменты кода, которые нужно использовать многократно для каких-либо вычислений. Функции могут работать с контекстом сессии: считывать или изменять его. Это позволяет объявить функцию один раз и затем вызывать ее из любого места сценария.

> **ПОДСКАЗКА**
>
> Рекомендуется добавлять блок **Скрипт** с описанием утилитарных функций в начало сценария или в препроцессинг. Во втором случае функции будут инициализироваться при каждом запросе.

Чтобы использовать утилитарные функции в проекте, опишите их в блоке **Скрипт**. После этого функции можно вызывать в других блоках проекта для чтения контекста сессии:

- **Текст**;
- **Кнопки**;
- **Динамические кнопки**;

- **HTTP-запрос**. В полях **Headers** и **Body** – для вычисления значений параметров запроса, в поле **Response mapping** – для записи значений переменных в зависимости от результата выполнения запроса;
- **LLM**. В полях **Системное сообщение** и **Пользовательское сообщение** – для вычисления значений параметров запроса к модели;
- **Ai-агент**. В полях **Системное сообщение** и **Пользовательское сообщение** – для вычисления значений параметров запроса к агенту;
- **Переменная** – для определения значений переменных, которые вычисляются с помощью языка Python;
- **Условие** – для получения контекста сессии из всех скоупов.

> **ВНИМАНИЕ**
>
> Функцию можно объявлять только как переменную верхнего уровня, например conext.session.func_you().
>
> В блоке **Скрипт** можно не только получить контекст для чтения, но и изменить его.

Пример использования утилитарной функции для чтения контекста

Предположим, в зависимости от возраста пользователя нужно выводить в чате фразу с местоимениями Вы, вы или ты. Например, если пользователю больше 18 лет, то обращаться к нему на «Вы»: «Если Ваш вопрос по номеру, с которого Вы сейчас обращаетесь, просто скажите «да».»

Для этого:

1. В рабочую область добавьте блок **Скрипт** для объявления утилитарных функций, если его еще нет. Пропишите в нем логику выбора обращения к пользователю в зависимости от возраста. Например:

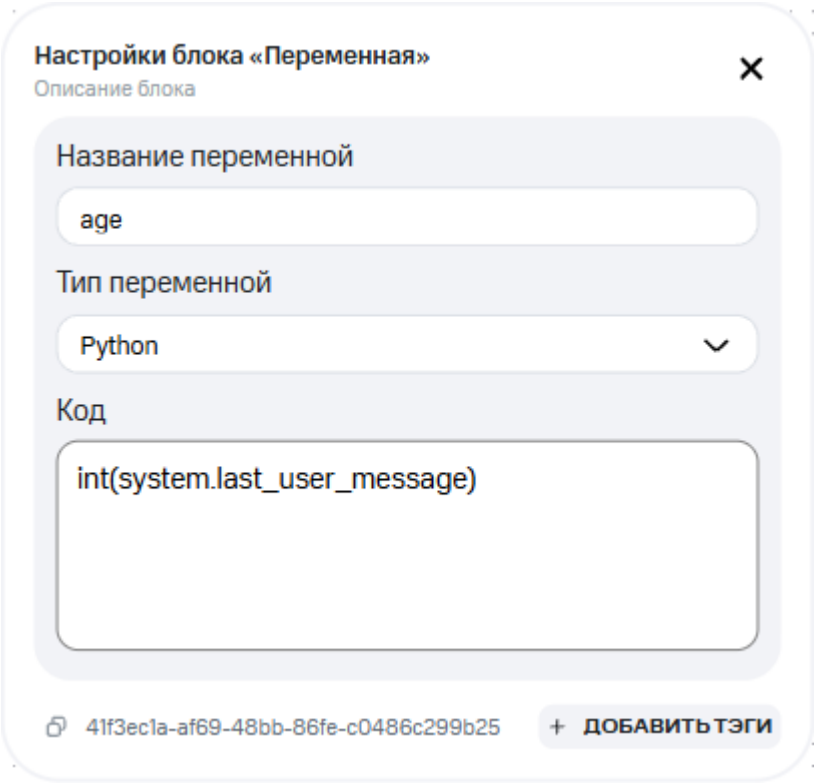
```
def func_you(adult, teen, child):
 """
 Возвращает фразу, соответствующую возрастной категории пользователя.

 :param adult: фраза для взрослых (18 лет и старше)
 :param teen: фраза для подростков (от 13 до 17 лет)
 :param child: фраза для детей (до 12 лет)
 :return: строка, соответствующая возрастной категории пользователя
 """
 if context.session.age >= 18:
     return adult
 elif 13 <= context.session.age < 18:
     return teen
 elif context.session.age < 13:
     return child

 # Функция обработчика - обязательная точка входа
 async def handler(context):
     context.session.func_you = func_you
```

> **ВНИМАНИЕ**
>
> В функции обработчика задайте переменную, в которую сохраняется результат работы функции. Например, context.session.func_you = func_you.

2. Добавьте текстовый блок, в котором будет задаваться вопрос о возрасте пользователя. После получения ответа сохраните эти данные в переменную, например age:



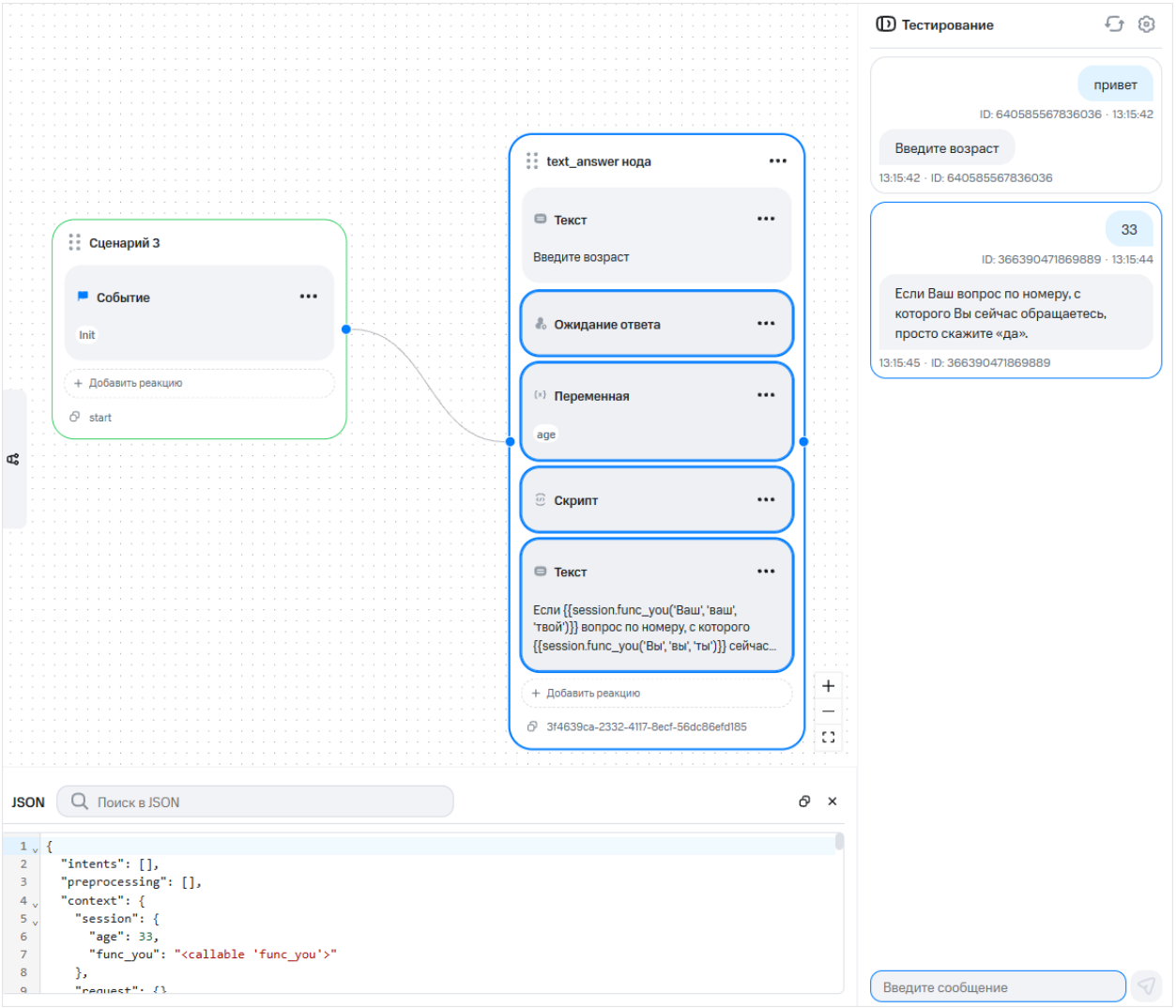
3. В нужное место сценария добавьте блок Текст для вывода ответа пользователю, в котором вызывается заданная функция. Например:

```
Если {{session.func_you('Ваш', 'ваш', 'твой')}} вопрос по номеру, с которого
{{session.func_you('Вы', 'вы', 'ты')}} сейчас {{session.func_you('обращаетесь',
'обращаетесь', 'обращаешься')}}, просто {{session.func_you('скажите', 'скажите',
'скажи')}} «да».
```

   Где: func_you() – функция, которая описана в реакционном блоке Скрипт, а также параметры, которая она получает на вход для обработки; {{}} – синтаксис использования переменных контекста; session – скоуп переменных, куда сохранена функция.

В результате объявленную утилитарную функцию можно вызывать в нужном месте сценария.

Пример работы сценария:



Пример использования утилитарной функции для записи в контекст

Предположим, нужно сохранить фамилию, имя и отчество пользователя в контекст сессии для дальнейшего использования в сценарии.

Для этого:

1. Добавьте блок **Скрипт** для объявления утилитарных функций, если его еще нет. Пропишите в нем нужную логику. Например, сохранение в переменную you последнего сообщения пользователя:

```
def func_save_personal_data(context):
 context.session.you = context.system.source_message

 async def handler(context):
 context.session.func_save_personal_data = func_save_personal_data
```

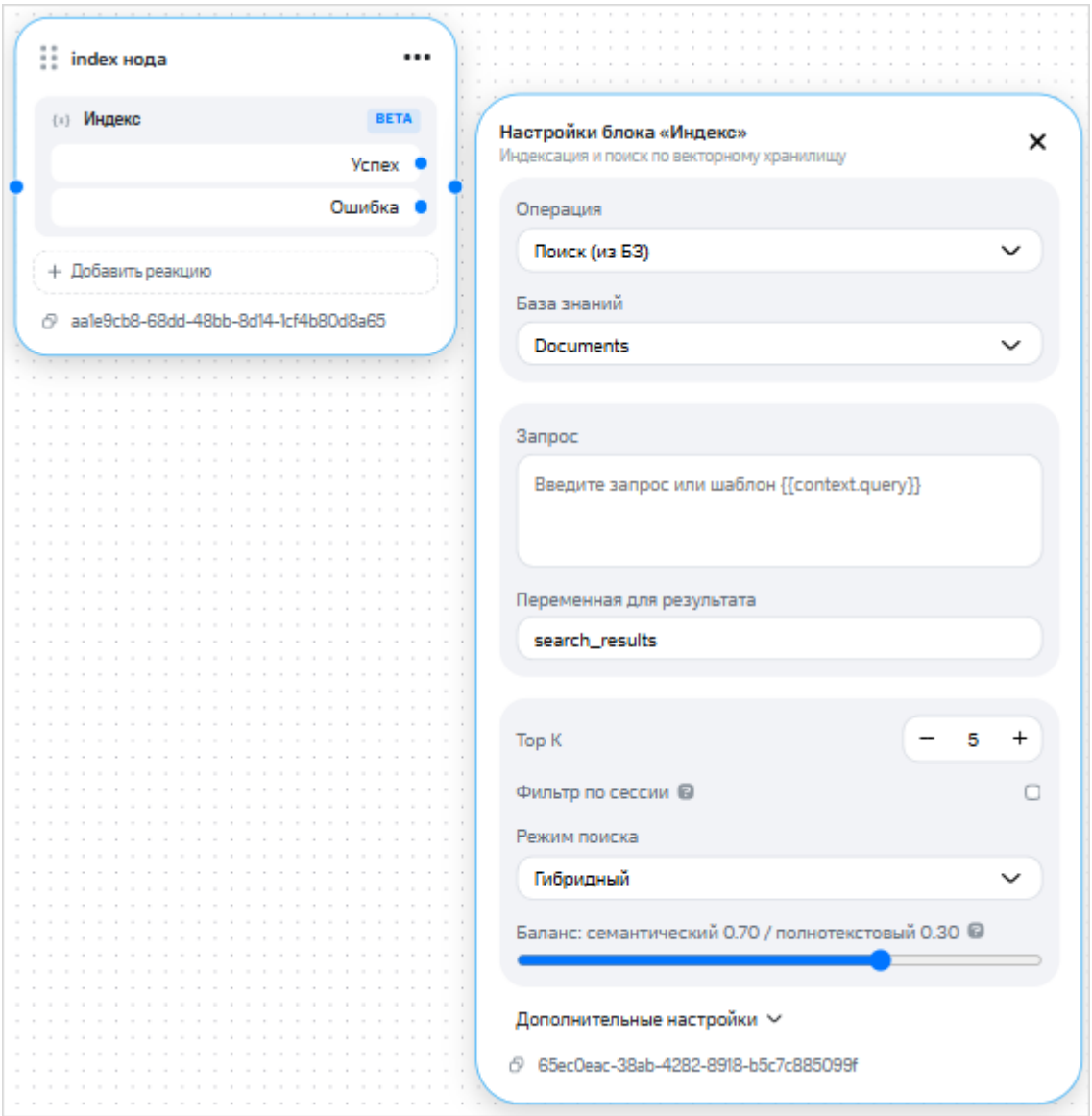
> **ВНИМАНИЕ**
>
> В функции обработчика задайте переменную, в которую сохраняется результат работы функции. Например, context.session.func_save_personal_data = func_save_personal_data.

2. В нужное место сценария добавьте текстовый блок, в котором задается вопрос о ФИО пользователя.
3. Добавьте блок **Скрипт**, в котором нужно вызывать ранее объявленную функцию, например:

```
# Функция обработчика - обязательная точка входа
def handler(context: Context) -> None:
    context.session.func_save_personal_data(context)
```

Индекс

Блок **Индекс** – центральный элемент пайплайна RAG в платформе AI Agents Platform.



Блок является точкой входа для работы с базой знаний RAG в сценариях. Блок может выполнять одну из двух операций:

- Загрузка чанков – добавление полученных на вход фрагментов документа (чанков) в индекс базы знаний.
- Поиск (из БЗ) – поиск в индексе релевантных фрагментов по запросу пользователя.

Подробнее см. в разделе «RAG в сценариях».

Параметры блока

Операция

Определяет режим работы блока – поиск или загрузка чанков. Набор остальных параметров зависит от выбранной операции: **Поиск (из БЗ)** или **Загрузка чанков**.

Поиск (из БЗ)

При выборе операции доступны параметры:

- **База знаний** – целевая база знаний, по которой выполняется поиск.
- **Поисковый запрос** – переменная с вопросом пользователя, обычно `{{system.source_message}}`.
- **TopK** – количество результатов поиска в выдаче. Рекомендуется 5.
- **Фильтр по сессии** – ограничивает поиск только чанками, загруженными в текущей сессии. Включение флажка гарантирует, что поиск выполняется только по документам, загруженным в рамках текущей сессии диалога с ботом. Это полезно при динамическом индексе, когда каждый пользователь загружает свои документы. При завершении сессии чанки автоматически удаляются.
- **Режим поиска** – тип поиска по индексу:
  - **Векторный** (по умолчанию) — семантический поиск по эмбеддингам, использует косинусное сходство. Наиболее эффективен для поиска по смыслу, когда слова в запросе не совпадают точно с документом.
  - **Полнотекстовый (BM25)** — поиск по ключевым словам. Ищет точное совпадение или близость терминов. Подходит для специфичных термов (команды, параметры, названия, ошибки).
  - **Гибридный** — комбинирует векторный и полнотекстовый поиск для получения лучших результатов. Результаты ранжируются с настраиваемым балансом весов. Оптимальный выбор для большинства случаев, когда база знаний содержит как описательные тексты, так и специфичные термины.
- **Баланс гибридного поиска** (появляется только при выборе режима «Гибридный») — ползунок для регулировки баланса между векторным и BM25 поиском:
  - **Значение** — от 0.0 до 1.0
  - **По умолчанию** — 0.70 (70% векторный, 30% BM25)
  - **Когда увеличивать** (> 0.70) — если документы содержат описательные тексты, синонимы, парафразы
  - **Когда уменьшать** (< 0.70) — если документы содержат много специфичных термов, кодов, названий команд, API-параметров

- **Переменная для результата** – имя переменной, куда блок сохранит найденные фрагменты.

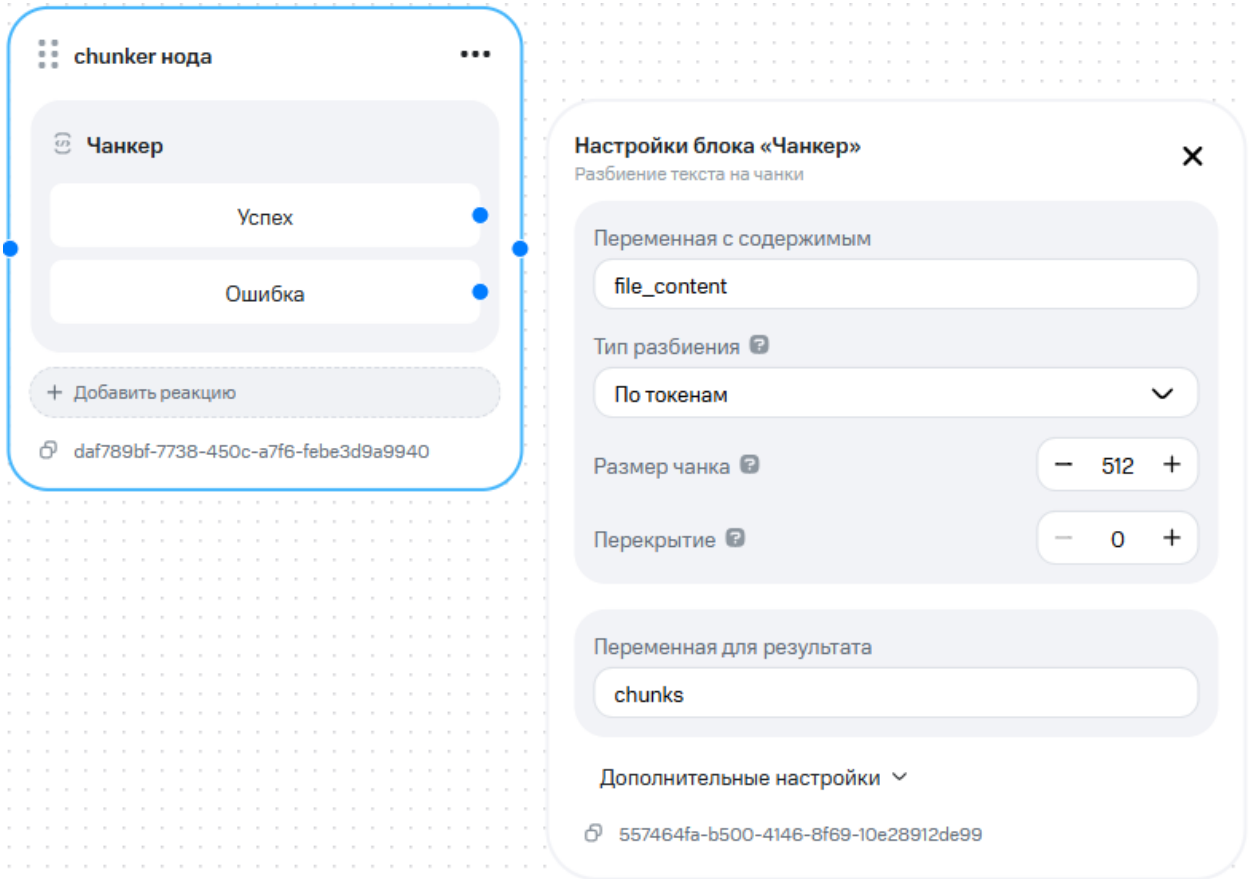
Загрузка чанков

При выборе операции доступны параметры:

- **База знаний** – целевая база знаний для загрузки чанков. Содержание базы значения не имеет – важны только параметры модели эмбеддера, выполняющей индексацию полученных чанков.
- **Переменная с чанками** – переменная, в которую блок **Чанкер** сохраняет фрагменты документа.
- **Переменная для результата** – имя переменной, куда блок сохраняет результаты индексации.

Чанкер

Блок разбивает текст на фрагменты (чанки) для последующего индексирования. Используйте этот блок в пайплайне RAG с динамическим индексом, когда пользователь загружает документ во время диалога с ботом. Подробнее о процессе см. в разделе «RAG в конструкторе сценариев».



Заполните параметры:

Переменная с содержимым. Имя переменной, из которой блок получает текст. Обычно это выходная переменная блока **Загрузчик файлов** – например, temp.file_content.

Тип разбиения. Определяет, как текст делится на фрагменты.

ЗНАЧЕНИЕ	ОПИСАНИЕ	ИСПОЛЬЗОВАНИЕ
По токенам	Разрезает текст каждые N токенов. Это значение установлено в блоке по умолчанию	Однородные данные: транскрипции звонков, выгрузки чатов, сырые данные и т.д.
По предложениям	Разрезает по границам предложений	Тексты, где важно сохранить смысл каждого предложения: FAQ, юридические и нормативные документы, контракты
Рекурсивное	Разрезает тексты иерархически, последовательно используя для этого все более мелкие разделители: заголовки, абзацы, строки, предложения. Игнорирует параметр **Перекрытие** – алгоритм самостоятельно определяет границы фрагментов на каждом уровне иерархии разделителей	Сложные, хорошо структурированные тексты с неоднородной структурой, в которых есть и длинные абзацы, и короткие списки: книги, исследовательские отчёты, техническая документация, статьи в Markdown

ПРИМЕЧАНИЕ

При выборе типа **Рекурсивное** параметр **Перекрытие** игнорируется — алгоритм самостоятельно определяет оптимальные границы фрагментов.

Размер чанка. Максимальное количество токенов в одном фрагменте. Влияет на точность и контекст поиска. Для блока **Чанкер** в сценариях рекомендуется использовать **512 токенов** – это обеспечивает баланс между детализацией и контекстом при индексации документов, загруженных пользователем.

ТИП ДОКУМЕНТА	РЕКОМЕНДОВАННЫЙ РАЗМЕР	ПРИМЕЧАНИЕ
FAQ	512 – 800	Однородные, краткие ответы
Руководства	1024 – 1200	Структурированные инструкции
Статьи	1200 – 1500	Развёрнутое содержание
Контракты	1500 – 2000	Сложный, многоуровневый текст
Большие документы	2048	Максимум для больших файлов

ПОДСКАЗКА

В блоке **Чанкер** для динамического индекса (загрузка документов пользователем во время диалога) типичные параметры: **размер 512, перекрытие 128 токенов**.

Перекрытие. Количество общих токенов между соседними фрагментами. Используется для сохранения контекста на стыках чанков. Рекомендуется устанавливать 15 – 20% от размера чанка (для размера 512 это примерно 128 токенов). Не используется при рекурсивном чанкировании.

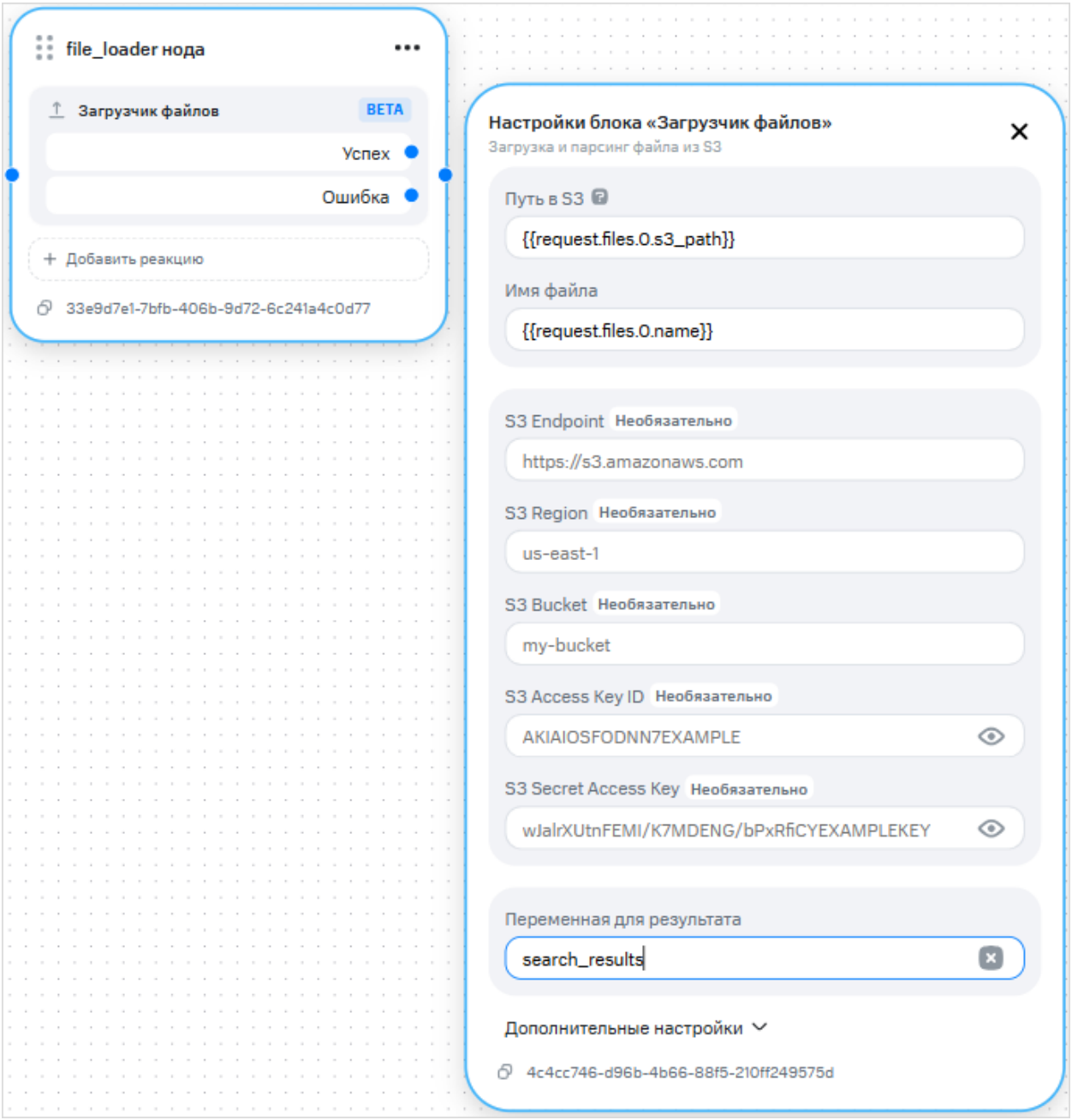
> **ВНИМАНИЕ**
>
> Значение перекрытия не должно превышать размер чанка. Максимальное допустимое перекрытие 30% от размера чанка.

Переменная для результата. Имя переменной для сохранения чанков. Укажите его же в поле **Переменная с чанками** блока **Индекс**. Подробнее о переменных см. в разделе «Переменные в сценарии».

> **ВНИМАНИЕ**
>
> Имя переменной для результата не должно совпадать с именем входной переменной. Если указать одинаковые имена, входные данные будут перезаписаны.

Загрузчик файлов

Блок Загрузчик файлов загружает документ из S3-хранилища, извлекает текст, сохраняет результат в переменную.



Заполните параметры:

Путь в S3. Явно прописанный путь (ключ) к объекту внутри бакета S3 или переменная, в которой хранится путь. По умолчанию задано – `{{request.files.0.s3_path}}`.

Имя файла. Переменная, содержащая имя загруженного файла. По умолчанию – `{{request.files.0.name}}`. Имя используется для определения парсера по расширению и передаётся в метаданные при дальнейшей индексации. Если в S3 файл хранится под служебным именем (например, UUID), здесь можно указать его оригинальное имя.

Переменная для результата. Имя переменной для сохранения извлечённого текста.

> **ВНИМАНИЕ**
>
> Если далее в сценарии используется блок **Чанкер**, то в его настройках в поле **Переменная с содержимым** укажите точно такое же значение переменной для результата.

Если файл загружается из S3-хранилища, отличного от системного, то дополнительно заполните параметры:

- **S3 Endpoint**. URL хранилища (MinIO, AWS S3, Yandex Cloud);
- **S3 Region**. Регион AWS. Только для AWS S3;
- **S3 Bucket**. Имя корзины;
- **S3 Access Key ID**. Ключ доступа;
- **S3 Secret Access Key**. Секретный ключ.

> **ВНИМАНИЕ**
>
> Если поля не заполнены, блок использует системные настройки.

Поддерживаемые расширения файлов:

- **Офисные документы** – DOC, DOCX, XLS, XLSX, XLT, XLTX, PPTX, PPT;
- **PDF-документы** – PDF;
- **Текстовые документы** – TXT, MD, CSV;
- **Веб-документы** – HTML, HTM;
- **Изображения** – JPG, GIF, PNG, WEBP (извлекаются как base64-строка; текст с картинок не распознаётся).

> **ПРИМЕЧАНИЕ**
>
> **RAG и изображения:** На текущем этапе RAG не работает с изображениями напрямую. Изображения можно загружать через **Загрузчик файлов**, но для их обработки необходимо используйте отдельные сценарии с использованием VLM для распознавания содержимого.

> **ПОДСКАЗКА**
>
> Для работы с RAG рекомендуется использовать блок **Загрузчик файлов** для извлечения текста, затем **Чанкер** для разбиения на фрагменты и **Индекс** для индексации в динамическом индексе БЗ.

:::note Особенности парсинга

- **Таблицы** (в DOCX, HTML, PPTX, XLSX) преобразуются в строки формата «Key: Value» по каждой строке таблицы.
- **Excel** (XLSX, XLS) – каждая строка извлекается с заголовками колонок.
- **PDF** поддерживается в двух режимах: `text` (быстрое извлечение) и `markdown` (анализ layout, медленнее).
- **Изображения** извлекаются как base64-строка, текстовая информация с картинок не распознаётся.

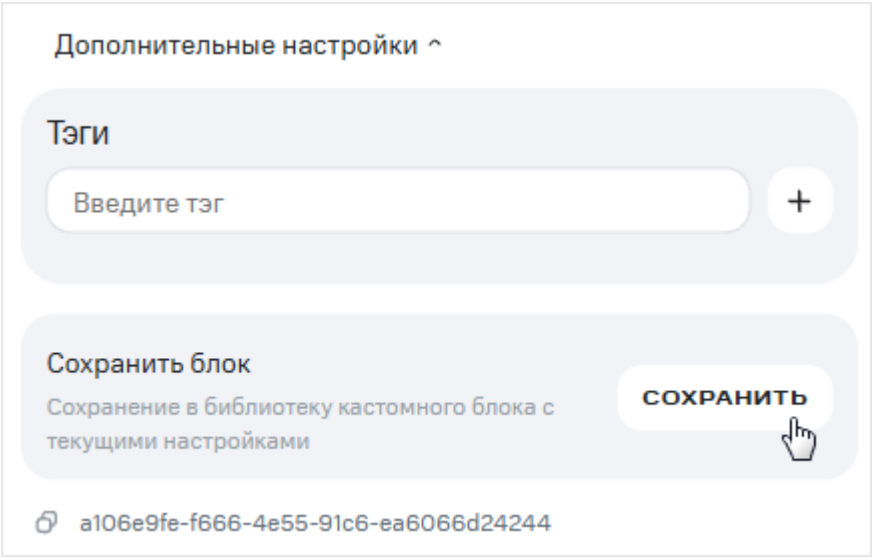
:::

Сохраненные блоки

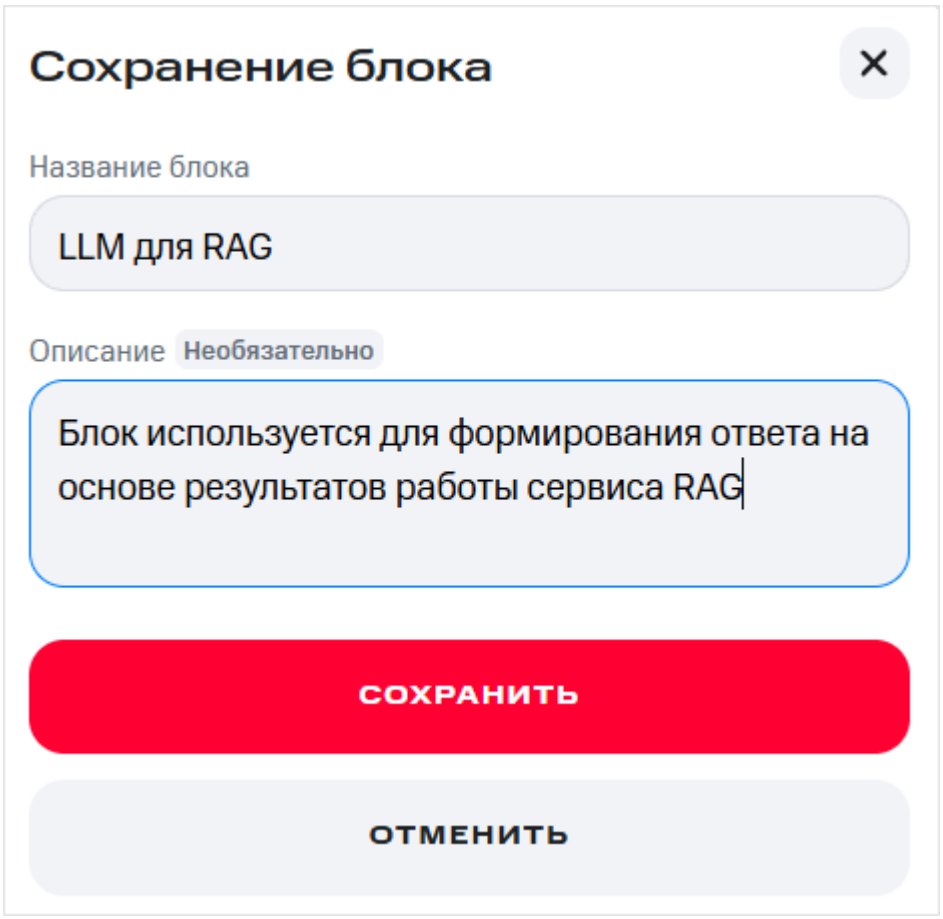
При разработке проекта некоторые идентично заполненные блоки нужно использовать в сценариях несколько раз. Например, запросы к LLM или агенту, интеграционные блоки **HTTP-запрос**. Для удобного переиспользования, чтобы не заполнять одни и те же значения повторно, блок можно сохранить. После этого его можно быстро добавить в рабочую область с панели блоков.

Чтобы сохранить блок:

1. Перейдите к нужному блоку с заполненными полями и разверните дополнительные настройки.
2. Нажмите на кнопку **Сохранить**.



3. Заполните название блока, оно должно быть уникальным для текущего проекта. Под заданным названием блок будет отображаться на панели блоков. При необходимости добавьте описание. Например:

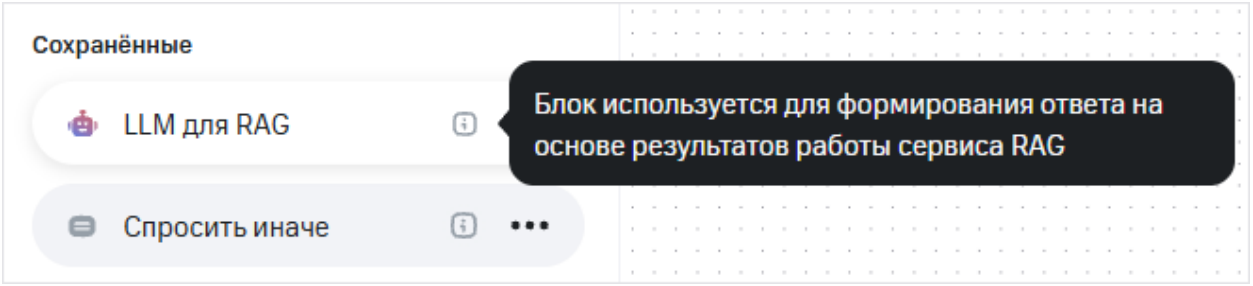


4. Нажмите на кнопку **Сохранить**.

В результате сохраненный блок отображается на панели **Блоки конструктора** в разделе **Сохраненные**. Он доступен для добавления в сценарии проекта, как и остальные блоки.

ПОДСКАЗКА

В качестве имени блока используется заданное при сохранении название, а при наведении на значок – открывается описание. Значок не отображается, если описание не было указано.



При необходимости значения полей можно скорректировать после переноса блока в рабочую область. При этом сам сохраненный блок скорректировать нельзя. Если блок не нужен, его можно удалить из раздела **Сохраненные**, в этом случае из сценариев он не удалится.

ВНИМАНИЕ

Сохраненный блок доступен только в текущем проекте. В других проектах нужно сохранять блоки заново.

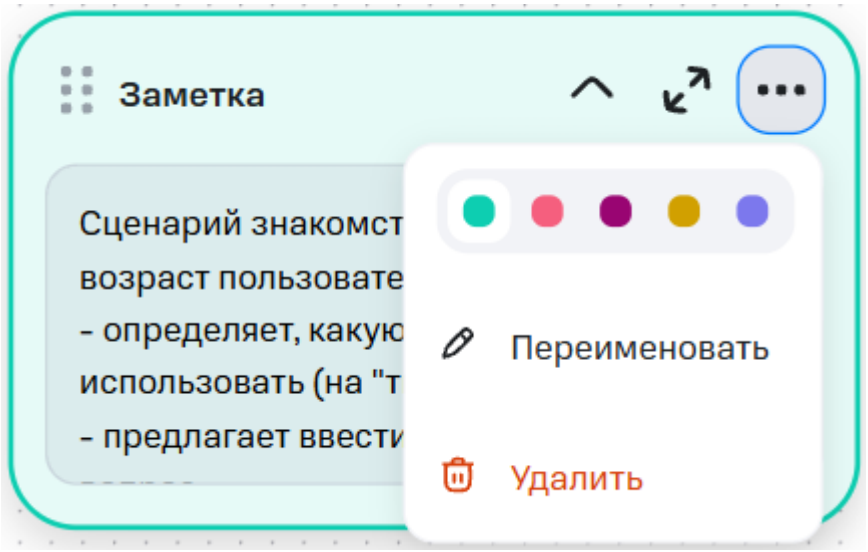
Заметки

В платформе к сценариям можно оставлять заметки. Например, кратко описать логику сложного сценария, чтобы другой пользователь конструктора мог быстро сориентироваться.

Заметка представляет собой отдельный блок, который можно разместить в любом месте сценария. Этот блок не участвует в логике, он является вспомогательным.

Чтобы добавить заметку:

1. Откройте панель **Блоки конструктора**.
2. Из раздела **Документация** перенесите блок **Заметка** в рабочую область.
3. Введите текст заметки и выберите цвет заметки:



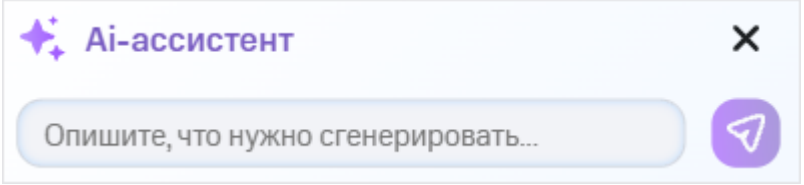
Для удобства по кнопке можно развернуть область для ввода текста.

При необходимости готовую заметку можно свернуть по кнопке , тогда блок принимает вид:

Чтобы переименовать или удалить блок, вызовите контекстное меню и выберите соответствующий пункт.

AI-ассистент для заполнения полей

Иногда заполнение полей с кодом, таких как Python-код и регулярные выражения, или полей с промптами занимает время и требует глубоких технических знаний. Чтобы оптимизировать этот процесс, в конструкторе сценариев есть возможность воспользоваться помощью AI-ассистента. Рядом с такими полями отображается кнопка . Нажмите на нее, чтобы открыть поле для ввода запроса ассистенту:



Введите запрос. В результате обработки AI-ассистент сгенерирует код, регулярное выражение или текст, который автоматически запишется в соответствующее поле.

> **ВНИМАНИЕ**
>
> Максимальное количество символов для ввода в поле AI-ассистента – 5000.

Если поле заполнено, то при генерации значения для него AI-ассистент может использовать контекст из этого поля. Результат сразу перезаписывает текущее значение, и при необходимости его можно отменить.

Системные промпты для заполнения этих полей уже заданы. Если качество генерации ответа вас не устраивает, то вы можете их изменить. Для этого используйте переменные с префиксом **AI_ASSIST_SYSTEM_PROMPT_** в файле values.yaml. Подробнее см. руководство администратора, раздел «Конфигурирование».

(пусто)

Переменные

В полях реакционных блоков можно использовать:

- зарезервированные переменные;
- переменные, созданные в сценарии;
- значения переменных для разных окружений, а также переменные с секретными значениями.

Формат использования переменных: **{{scope.name}}**

Где:

- **scope** – область видимости;
- **name** – имя переменной.

Зарезервированные переменные

В платформе предусмотрены системные переменные, значения которых по умолчанию рассчитываются движком. Их можно использовать при наполнении сценария логикой. Например, при составлении выражения в блоке **Условие** укажите переменную **last_user_message**, чтобы проверить последнее сообщение от пользователя.

> **ВНИМАНИЕ**
>
> При обращении к предопределенным переменным указывайте область видимости. Например, для системных переменных префикс **system**. Формат: **<Префикс>.<Имя переменной>**. Например, **system.last_user_message**. Если префикс не указан, то переменной автоматически присваивается префикс по умолчанию – **session**.

system

ПЕРЕМЕННАЯ	ТИП	ОПИСАНИЕ
system.system_session_id	STRING	Идентификатор чата
system_message_id	STRING или INT	Идентификатор запроса пользователя
sure_topic	STRING	Последний определенный интент в диалоге пользователя с ботом. Заполняется автоматически при любом запросе
topic_score	DOUBLE	Вероятность **sure_topic**
last_error_message	STRING	Последнее сообщение об ошибке с локализацией места в сценарии, где она произошла
last_user_message	STRING	Последнее сообщение пользователя из диалога с ботом. **Примечание**. Текст сообщения сохраняется в нижнем регистре
united_user_messages	STRING	Конкатенация (объединение) всех сообщений пользователя в рамках диалога
last_user_message_length	INT	Количество символов в последнем сообщении пользователя из диалога
messages_number	INT	Номер сообщения пользователя в диалоге
number_of_words	INT	Количество слов в последнем сообщении пользователя из диалога
source_message	STRING	Текст сообщения **Примечание**. В тексте сообщения сохраняется исходный регистр
surfac_metadata	JSON OBJECT	Текст сообщения **Примечание**. В тексте сообщения сохраняется исходный регистр
surface_metadata	JSON OBJECT	Метаданные поверхности. Содержит данные поверхности, которые могут быть использованы в сценарии. Например, system.surface_metadata.chat_id, system.surface_metadata.user_id, system.surface_metadata.fields
utc_now_dt	STRING	Текущие дата и время
response_additional_data	JSON OBJECT	Дополнительные данные для ответа. Значение задается вручную в блоке **Скрипт**. Пример: def handler(context: Context) -> None: context.system.response_additional_data = {"step": 1}
channel_id	STRING	ИД канала

ПЕРЕМЕННАЯ	ТИП	ОПИСАНИЕ

input	JSON OBJECT	Данные входящего сообщения.
		В зависимости от типа сообщения объект имеет вид:
		{
		"type": "message",
		"original_text": "text"
		}
		или
		{
		"type": "event",
		"name": "some_event",
		"data": {"data_1":"data_1_value"},
		}
		Данные из переменной можно использовать для построения логики. Например, если пришло входящее сообщение с типом "event", то можно активировать сценарий по событию с именем "some_event". Подробнее об активации по пользовательскому событию см. раздел «Событие»

nlu

Область видимости создается автоматически. Переменные заполняются результатами работы классификатора или регулярных выражений.

ПЕРЕМЕННАЯ	ТИП	ОПИСАНИЕ
matches	ARRAY	Кандидаты для активации на основе регулярных выражений. Вычисляются движком автоматически до препроцессинга
intents	ARRAY	Кандидаты-интенты для активации. Не вычисляются движком автоматически. Чтобы вычислить их, нужно вызвать контекстную функцию **predict_intent** в блоке **Скрипт** сценария препроцессинга
raw_intents	ARRAY	Список интентов и их вероятности (score), сформированный в результате анализа сообщения. Интент и score первого элемента списка также сохраняется в переменные **system.sure_topic** и **system.topic_score** соответственно

При необходимости в переменных можно переопределять результаты работы классификатора. Для этого в блоке **Скрипт** добавьте код, например:

```
def handler(context: Context) -> None:
    edge = IntentEdge(type="intent", value="new", threshold=0.5,
target_node_id="node2")
    new_candidate = ActivationCandidate(scenario_id=2, edge=edge, score=0.8)
    context.nlu.intents.append(new_candidate)
```

events

ПЕРЕМЕННАЯ	ТИП	ОПИСАНИЕ
candidates	ARRAY	Массив кандидатов для активации по событию

Области видимости и времени жизни переменных

Созданную пользователем или зарезервированную системой переменную можно определить в одну из областей видимости и времени жизни – scope (скоуп).

> **ВНИМАНИЕ**
>
> Если переменная используется в сценарии и префикс для нее не указан, система автоматически относит переменную в session-скоуп и присваивает префикс **session**. Поэтому при обращении к зарезервированным переменным указывайте префикс.
>
> Если имя переменной совпадает с названием скоупа, система автоматически разрешает конфликт, чтобы избежать неоднозначности. Например, имя **session** преобразуется в **session.session**.
>
> При использовании переменных в блоке Скрипт нужно дополнительно указать префикс **context**. Пример: **context.session.value**

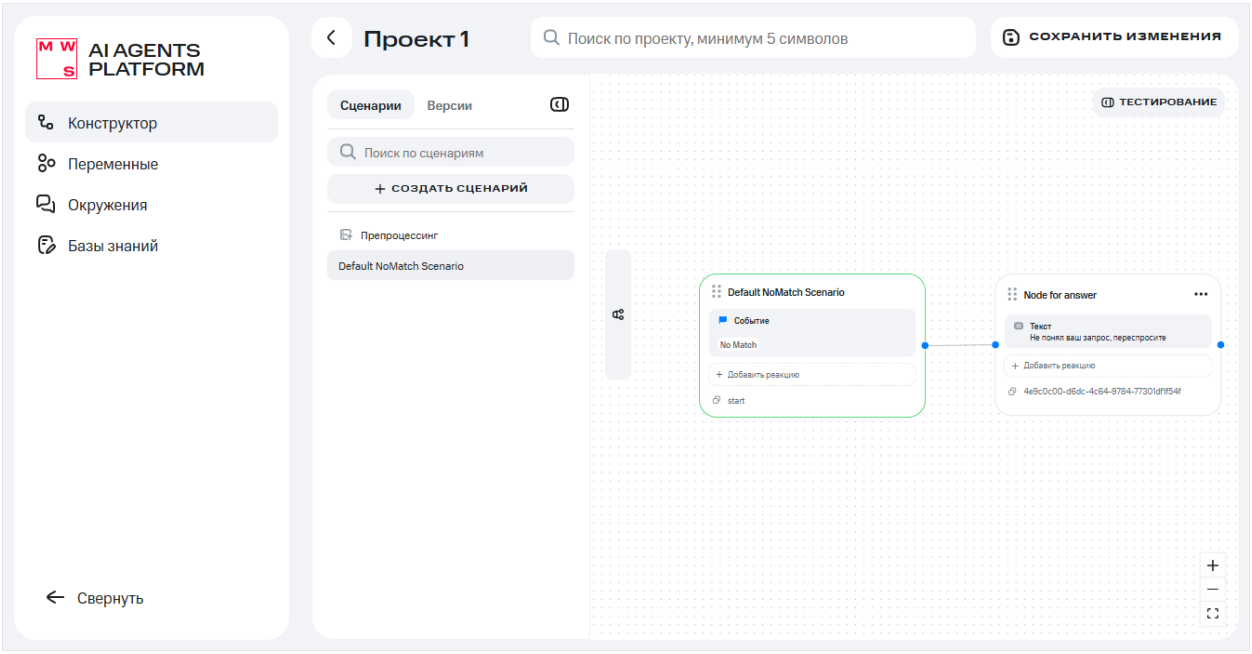
ОБЛАСТЬ ВИДИМОСТИ	ОПИСАНИЕ	ПРАКТИЧЕСКОЕ ИСПОЛЬЗОВАНИЕ	ПРИМЕР
session	Может использоваться пользователем для записи. Время жизни – пока диалоговая сессия активна	Долгосрочное хранение данных. Бот «запоминает» контекст, делая взаимодействие естественным. Пользователю не нужно повторять информацию. Используется, например, для персистентных данных клиента: можно записать в переменную список симптомов из предыдущих сообщений пациента. При новом цикле «реплика-ответ» в рамках текущей сессии бот проанализирует список из переменной и предложит рекомендации. Снижает нагрузку на пользователя , при этом данные хранятся только в сессии, что важно для приватности. Длительность сессии задается пользователем при создании канала, подробнее см. раздел «Публикация в канале»	session.value

ОБЛАСТЬ ВИДИМОСТИ	ОПИСАНИЕ	ПРАКТИЧЕСКОЕ ИСПОЛЬЗОВАНИЕ	ПРИМЕР
request	Может использоваться пользователем для записи. Время жизни — в пределах одной обработки запроса: от реплики пользователя до ответа бота	Для временных данных текущего запроса. Бот «забывает» данные из переменной после ответа на запрос клиента. Например, пациент описывает в сообщении симптомы, бот анализирует их и дает ответ, после чего данные стираются. Для оптимизации производительности ненужные данные не накапливаются	request.value
temp	Может использоваться пользователем для записи. Время жизни ограничено рамками выполнения сценария с восстановлением после прерываний	Хранение данных для промежуточных шагов сценария. Устойчивы к прерываниям сценария. Если бот выполняет блоки сценария Ожидание ответа или Переход в сценарий, переменная не стирается. Создается snapshot (снимок состояния) и при возобновлении сценария, данные восстанавливаются в точности такими, какими были до прерывания. Например, в при опросе пациента ботом симптомы сохраняются в переменную. Бот запрашивает у клиента уточнения. Если пациент не отвечает сразу, сценарий прерывается. После продолжения данные переменной восстанавливаются и бот продолжает, используя старые ответы из переменной. Для клиента экономится время — не нужно повторять ответы. Для системы эти же данные могут быть, например, интегрированы в отчет по окончании диалога. Полезно применять в ветвлениях графа, данные не теряются, даже если процесс приостанавливается	temp.value
system	Недоступен для записи данных. Время жизни переменных управляется системой	Скоуп системных переменных	system.last_user_message

ОБЛАСТЬ ВИДИМОСТИ	ОПИСАНИЕ	ПРАКТИЧЕСКОЕ ИСПОЛЬЗОВАНИЕ	ПРИМЕР
nlu	Создается автоматически, заполняется результатами работы классификатора или регулярных выражений. При необходимости список кандидатов можно изменить	Чтобы вычислить кандидатов для активации по правилам или интентам, нужно добавить программный код в сценарий препроцессинга. Подробнее о функциях см. в описании блока Скрипт, раздел «Скрипт». При необходимости в переменные можно добавлять нового кандидата. Для этого также необходимо добавить код в блоке **Скрипт**	nlu.matches
events	Создается автоматически при обработке входящего события	Заполняется информацией о кандидатах для активации по событию	events.events_candidates
env	Создаются пользователем для использования в окружениях	Переменные предназначены для использования в окружениях, например, если в сценарии нужно использовать разные значения для тестового и продуктового контура. Также в переменных можно сохранять секретные значения, такие как токены, чтобы они не были видны	env.URL_api

Переменные и окружения

В платформе можно создавать переменные, которыми заполняются поля реакционных блоков. В зависимости от окружения значения этих переменных в сценариях меняется. Управление переменными и окружениями выполняется в соответствующем разделе на панели слева в конструкторе сценариев. По умолчанию панель свернута. Чтобы развернуть ее, нажмите на кнопку →.



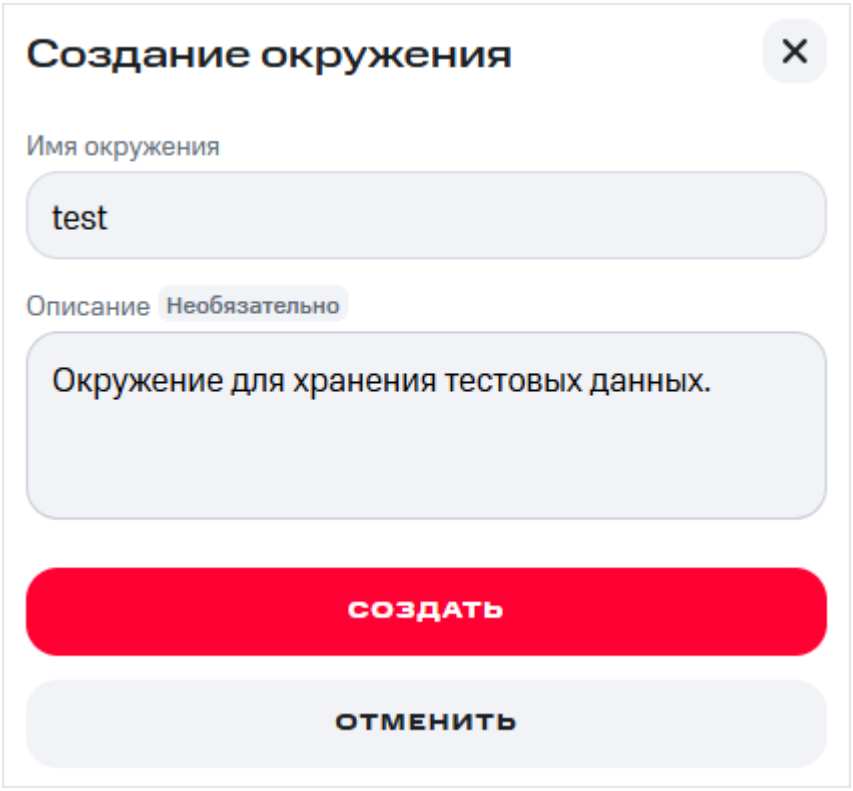
Предположим, в сценарии нужно выполнять HTTP-запрос по разным URL для тестового и продуктового контура. Также требуется защитить секретные данные, такие как токены. Для этого создайте окружения test и prod. Кроме этого, создайте переменную URL_api и запишите в нее URL-адреса для использования в продуктовом и тестовом контуре, а в переменную Token — сохраните значение токена для вызова LLM.

Чтобы использовать переменные и окружения в проекте:

1. Создайте окружения.
2. Создайте переменные и установите их значения для одного из окружений.
3. При необходимости настройте значения переменных для остальных окружений.
4. Добавьте переменные в сценарий.
5. Протестируйте проект со значениями каждого из окружений.
6. При публикации проекта в канале выберите нужное окружение.

Создание окружений

1. Перейдите в раздел **Окружения**.
2. Нажмите на кнопку **Создать окружение**.
3. Заполните поля:

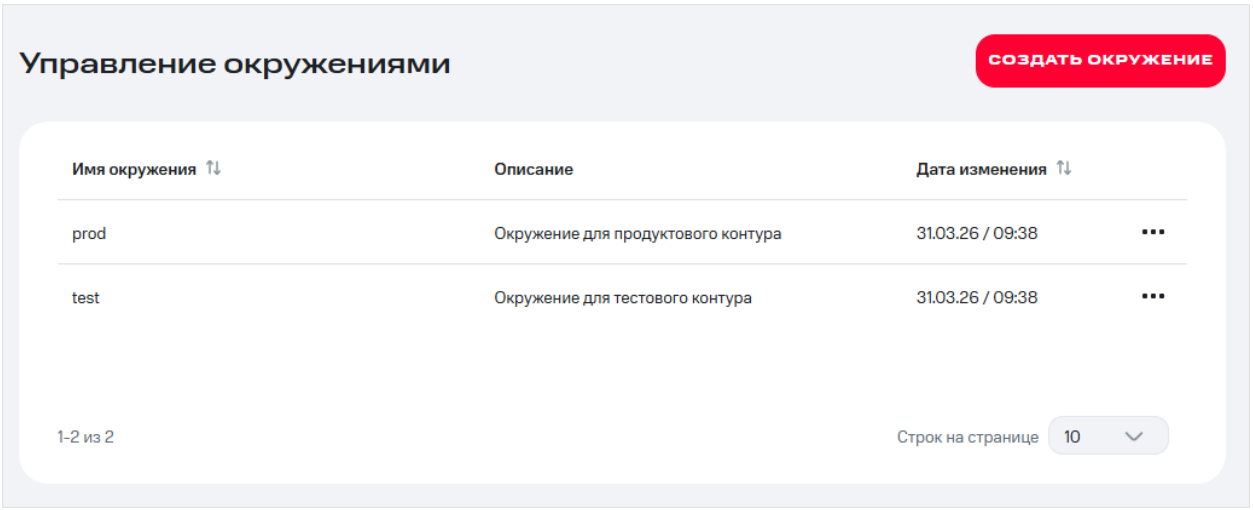


Имя окружения. Задайте имя окружения. Допускается от 4 до 16 символов.

Описание. Добавьте описание, если необходимо. Оно будет отображаться в списке окружений.

1. Нажмите на кнопку **Создать**.

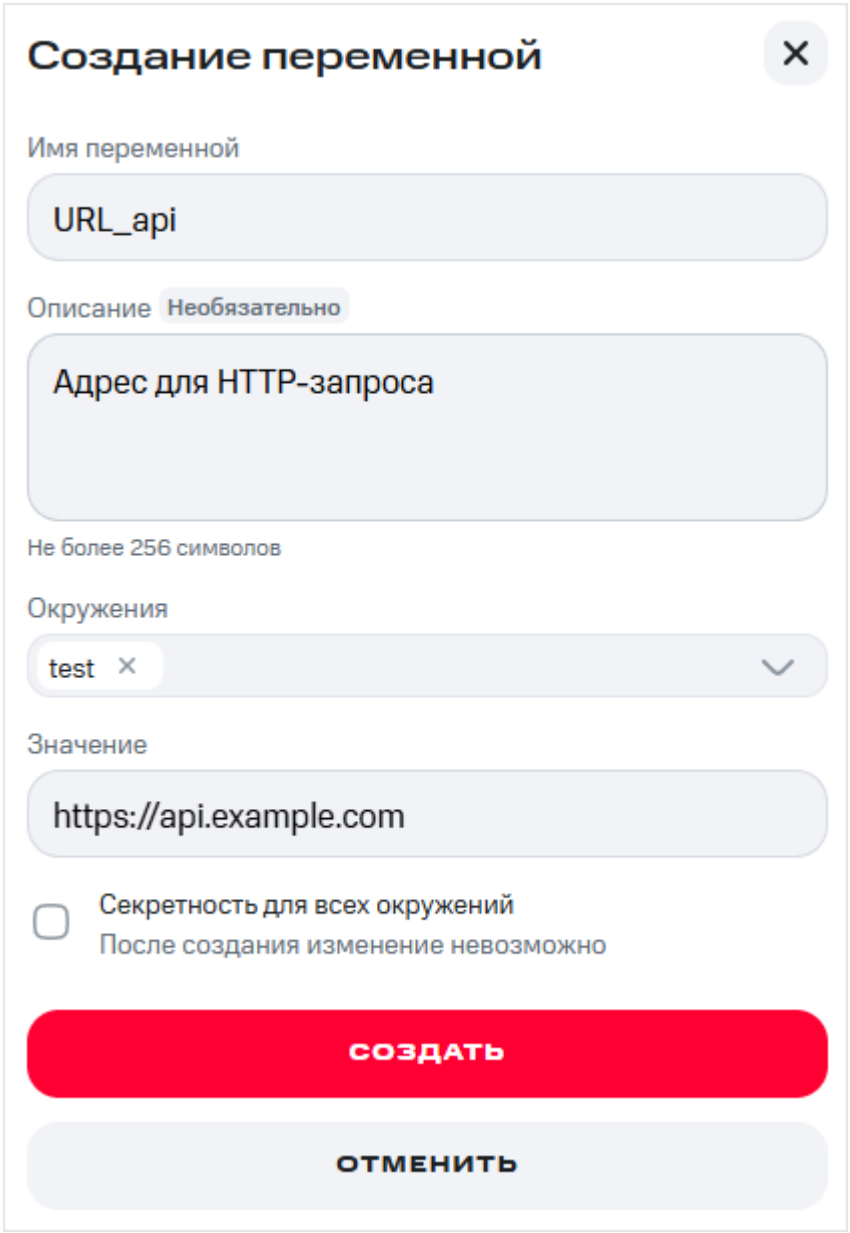
В результате созданные окружения отображаются в списке:



При необходимости вы можете изменить настройки окружений. Также окружения можно удалить.

Создание переменных

1. Перейдите в раздел **Переменные**.
2. Нажмите на кнопку **Создать переменную**.
3. Заполните поля:



Имя переменной. Укажите имя переменной. Допускается от 4 до 64 символов.

Описание. Добавьте описание, если необходимо. Оно будет отображаться в списке переменных.

Окружения. Выберите окружения, для которых созданная переменная будет принимать указанное значение.

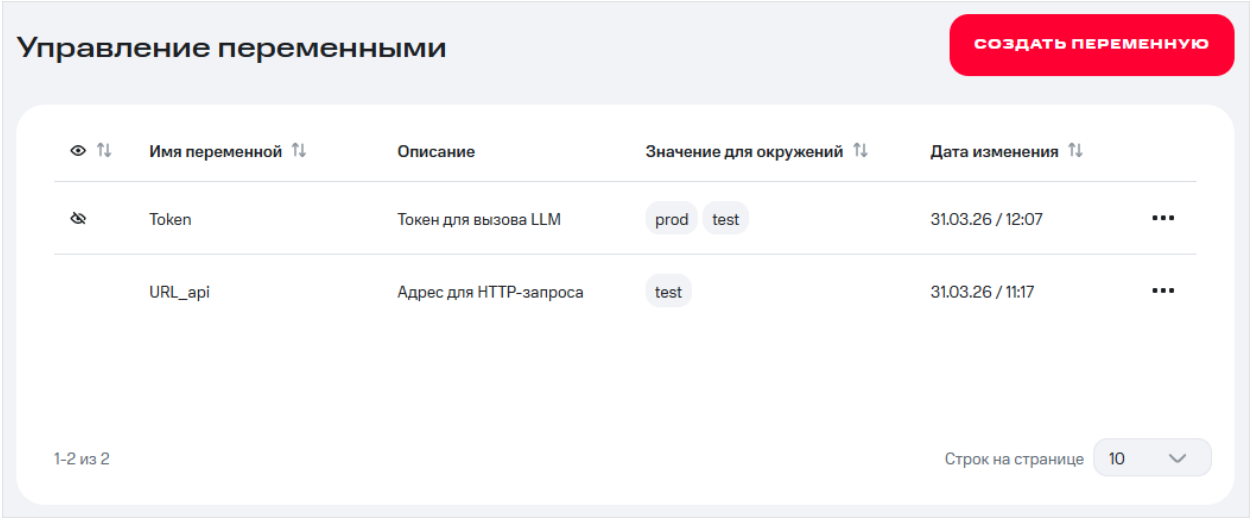
Значение. Укажите значение, которое должна принимать переменная для выбранного окружения. Значение должно быть указано хотя бы для одного окружения.

Секретность для всех окружений. Установите флажок, если нужно скрыть значение переменной.

> **ВНИМАНИЕ**
>
> Если при создании переменной флажок не установлен, то это можно сделать позднее. Если секретность указана, то изменить этот параметр позже нельзя.

4. Нажмите на кнопку **Создать**.

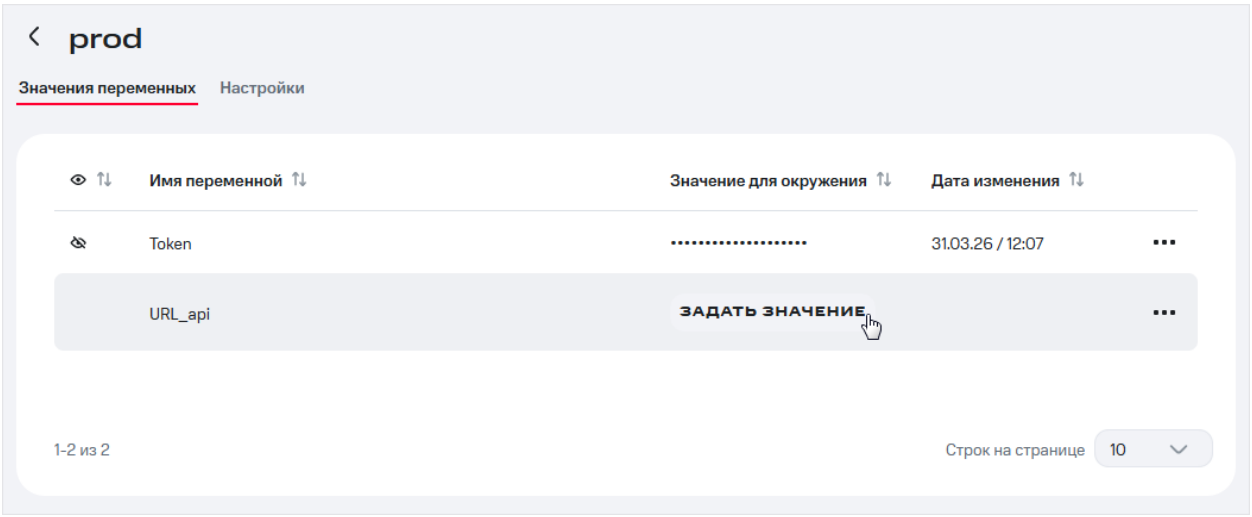
Аналогично создайте все необходимые переменные. Они отображаются в разделе **Переменные**. В столбце **Значение для окружений** указаны окружения, для которых определено значение созданных переменных.



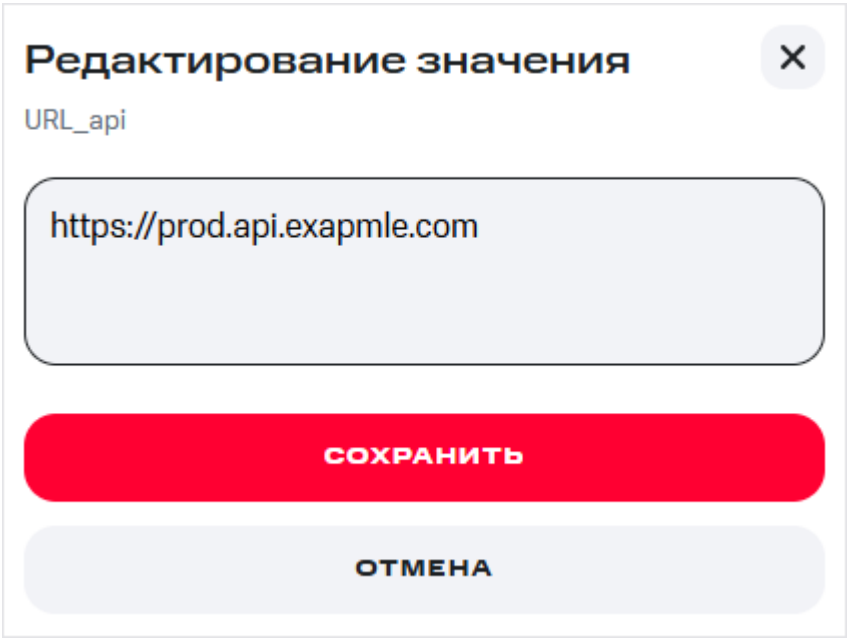
Настройка переменных для окружений

Чтобы в различных окружениях созданные переменные принимали свои значения:

1. В разделе **Окружения** по кнопке ⋯ вызовите контекстное меню. Выберите пункт **Перейти в настройки**.
2. В строке переменной, для которой нужно изменить значение, нажмите на кнопку Задать значение:



1. Укажите значение для выбранного окружения.



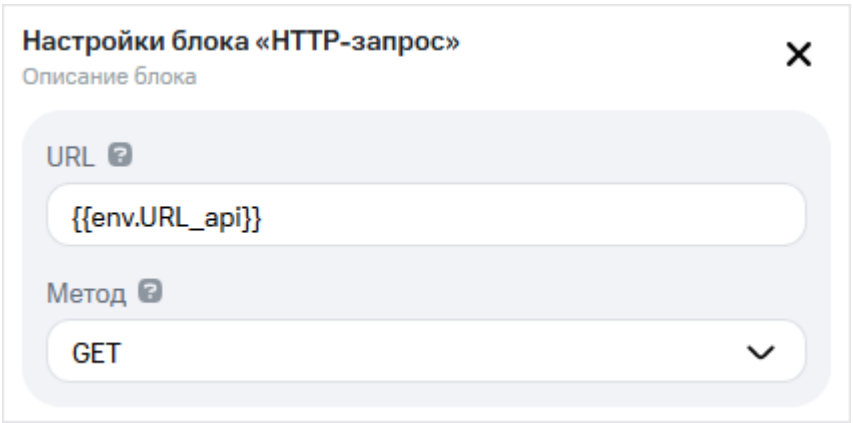
1. Нажмите на кнопку **Сохранить**.

В результате значение для окружения сохраняется и отображается в списке. Если для окружения не требуется специфическое значение переменной, то оставьте его незаполненным.

Использование переменных в сценариях

При заполнении полей блоков конструктора переменными используйте шаблон: {{env.<Имя переменной>}}, где env – область видимости.

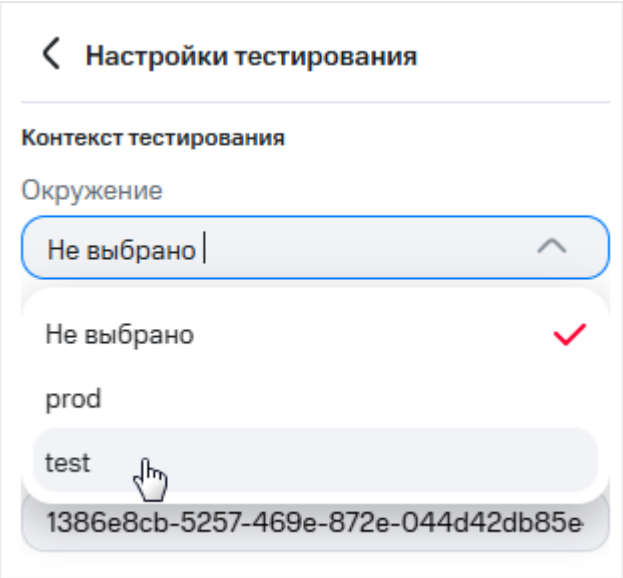
Пример обращения к переменной в блоке HTTP-запрос:



Тестирование проекта с переменными окружений

Если в ваших сценариях используются разные значения переменных для окружений, то перед тестированием выберите окружение.

Для этого перейдите в настройки тестирования и в выпадающем списке **Окружение** выберите одно из значений:



В результате при тестировании в переменные подставляются значения из указанного окружения.

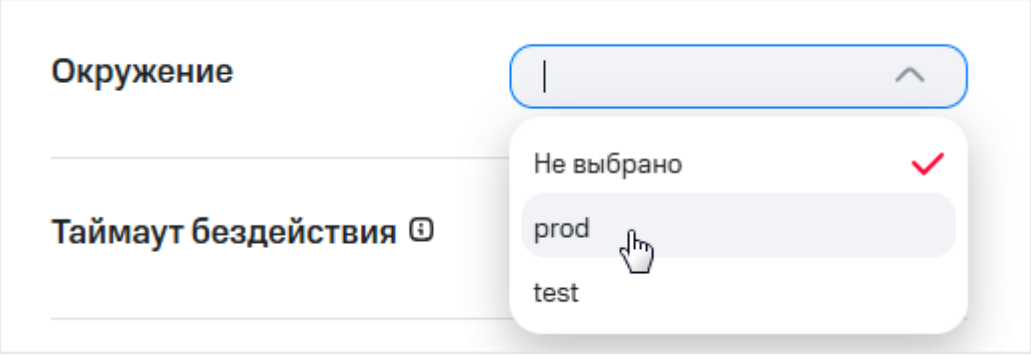
> **ВНИМАНИЕ**
>
> Если окружение не выбрано, то поле, которое заполнено переменной окружения, остается пустым. Во время прохождения сценария возникает ошибка.

После успешного тестирования опубликуйте версию, чтобы она стала доступна для публикации в канале.

Выбор окружения в настройках канала

Если в проекте используются переменные окружений, то при публикации в канале нужно выбрать окружение.

1. Перейдите в настройки канала.
2. Выберите проект.
3. Выберите окружение в соответствующем поле:



1. Сохраните изменения.

(пусто)

RAG

RAG (Retrieval-Augmented Generation) – способ генерации ответов Языковой моделью (LLM) с опорой на внешнюю базу знаний, когда перед формированием ответа модель получает релевантный контекст, извлечённый из документов, хранящихся в базе.

LLM обучена на больших объёмах данных, но не имеет онлайн доступа к конкретным ресурсам: внутренним документам, регламентам, инструкциям, справочникам и т.д. RAG решает эту проблему:

- бот отвечает на вопросы, используя релевантные данные, а не общие знания модели;
- ответы подкрепляются фактами и ссылками на документы базы знаний;
- базу знаний обновляется без переобучения модели – достаточно добавить или заменить документы.

Принцип работы

RAG в базовом варианте состоит из двух связанных этапов: индексации документов и поиска по ним при ответе на вопрос.

Индексация – подготовка документов, загруженных в базу знаний, к поиску, включает три стадии:

- Парсинг – из файла (PDF, DOCX, TXT и др.) извлекается текст.
- Чанкинг – текст разбивается на фрагменты (чанки) по структуре или по заданному размеру.
- Эмбеддинг – каждый фрагмент преобразуется в числовой вектор (эмбеддинг), представляющий смысл текста. Все эмбеддинги сохраняются в индекс – структуру, по которой быстро и удобно выполняется поиск. Индекс сохраняется в векторную базу данных.

Процесс индексации: Документ → Парсинг → Чанки → Эмбеддинги → Индекс

Поиск и генерация ответа – выполняются при каждом вопросе пользователя:

- Вопрос пользователя проходит эмбеддинг.
- В индексе находятся фрагменты, векторы которых приближаются по значению к эмбеддингу вопроса.
- Найденные фрагменты передаются языковой модели в качестве контекста.
- Модель генерирует ответ, опираясь на найденные фрагменты.

Процесс поиска и генерации ответа: Вопрос → Эмбеддинг вопроса → Поиск по индексу → Контекст → LLM → Ответ

Поиск работает по смыслу, а не по точному совпадению слов. Векторные представления позволяют находить релевантные фрагменты даже, если в них отсутствуют слова из формулировки вопроса.

Блоки RAG-пайплана

В конструкторе сценариев этапы RAG реализуются с помощью специализированных блоков реакции:

БЛОК	ЭТАП RAG	ЧТО ДЕЛАЕТ
Загрузчик файлов	Парсинг	Загружает документ из хранилища и извлекает из него текст

БЛОК	ЭТАП RAG	ЧТО ДЕЛАЕТ
Чанкер	Чанкирование	Разбивает текст на фрагменты заданного размера с перекрытием
Индекс (режим **Загрузка чанков**)	Эмбеддинг чанков + сохранение в индекс	Преобразует фрагменты в векторы и сохраняет в индекс
Индекс (режим **Поиск**)	Поиск по индексу	Ищет в индексе фрагменты, ближайшие по смыслу к запросу
LLM	Генерация ответа	Генерирует ответ на основе найденных фрагментов

Подробное описание параметров каждого блока см. в разделах «Загрузчик файлов», «Чанкер», «Индекс».

Помимо специализированных блоков вы можете использовать в RAG-пайплайне любые другие блоки конструктора: **Переменная**, **Условие**, **Скрипт**, **Текстовый ответ**, **HTTP-запрос**, **Ожидание ответа** и другие. Подробнее см. раздел «Расширение RAG-пайплайна».

Варианты RAG-пайплайна

В качестве примера приведем два варианта RAG-пайплайна. Они различаются тем, когда и как выполняется индексация — то есть на каком этапе документы проходят через цепочку «парсинг → чанкинг → эмбеддинг» и попадают в индекс.

RAG со статическим индексом. Индексация выполняется заранее — на этапе проектирования бота. Вы загружаете документы в базу знаний через интерфейс платформы, и платформа автоматически индексирует их в фоновом режиме. В сценарии используется только блок **Индекс** в режиме поиска — вся подготовка документов уже выполнена. Работа с базами знаний описана ниже.

Порядок работы: Индекс (поиск) → Подготовка контекста → LLM → Ответ

RAG с динамическим индексом. Индексация выполняется «на лету» — прямо в процессе диалога с ботом. Пользователь загружает документ в чат, и сценарий самостоятельно выполняет всю цепочку: загрузка файла, разбиение на фрагменты, сохранение в индекс. Данные документа привязаны к текущей сессии пользователя и автоматически удаляются при её завершении. То есть другие пользователи не увидят этот документ в ответах на их запросы.

Порядок работы: Загрузчик файлов → Чанкер → Индекс (Загрузка чанков) → Ожидание вопроса → Индекс (поиск) → Подготовка контекста → LLM → Ответ

Блок **Индекс** может получить чанки и из других источников — например, блока HTTP-запрос.

Сравнение вариантов

ХАРАКТЕРИСТИКА	СТАТИЧЕСКИЙ RAG	ДИНАМИЧЕСКИЙ RAG
Время индексации	На этапе проектирования сценария	В процессе диалога с ботом

ХАРАКТЕРИСТИКА	СТАТИЧЕСКИЙ RAG	ДИНАМИЧЕСКИЙ RAG
Источник документов	Загрузка через интерфейс платформы	Пользователь прикрепляет файл в чате
Время жизни данных	Ограничено только потребностями пользователей, могут быть обновлены или удалены администратором	В рамках сессии, удаляются при завершении
Блоки в сценарии	Индекс (поиск) → LLM	Загрузчик → Чанкер → Индекс (загрузка) → Индекс (поиск) → LLM
Область поиска	Вся база знаний	Документы текущей сессии
Видимость документов	Видны в интерфейсе базы знаний	Не отображаются в интерфейсе
Управление документами базы	Через интерфейс платформы	Автоматическое, привязано к жизненному циклу сессии

Варианты реализации

- Создание RAG со статическим индексом — для фиксированной БЗ
- Создание RAG с динамическим индексом — для документов, загружаемых пользователем
- Интеграция RAG-пайплайнов в сценарии — примеры использования
- Расширение RAG-пайплайна — продвинутые техники

Пошаговые руководства по созданию каждого варианта см. в разделах выше.

Базы знаний

База знаний – хранилище документов для RAG. Содержит загруженные документы и их векторные представления (эмбеддинги), собранные в индекс – структуру, по которому выполняется поиск.

Роль базы знаний в RAG

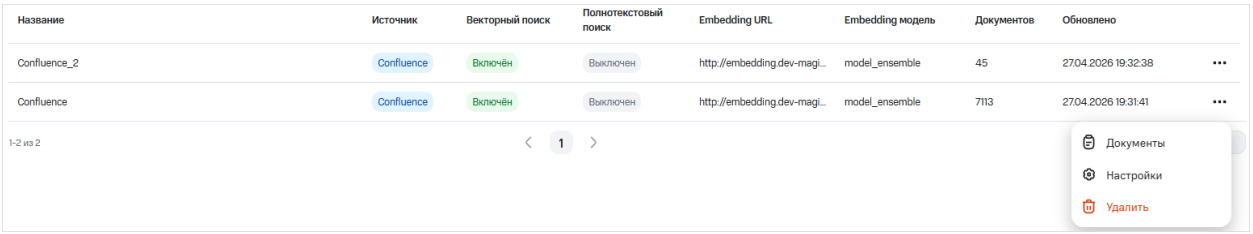
Для RAG со статическим индексом база знаний – основной источник данных. Вы создаёте базу заранее, загружаете документы, и платформа индексирует их автоматически в фоновом режиме. Блок **Индекс** в сценарии выполняет только поиск по готовой базе. Все пользователи одного бота ищут ответы в одной и той же базе знаний.

Для RAG с динамическим индексом база знаний также необходима – блок **Индекс** в режиме загрузки чанков использует её модель эмбеддинга для индексации документов, загруженных пользователем во время диалога. При этом документы и индексы каждого пользователя изолированы на уровне диалоговой сессии и автоматически удаляются при её завершении.

Список баз знаний

В проекте может быть несколько баз знаний – например, отдельные базы для разных тематик или продуктов. При настройке блока Индекс в сценарии выбирается, к какой именно базе обращаться для поиска или загрузки чанков.

Все базы знаний проекта отображаются в разделе **База знаний** в виде списка, отсортированного по дате последнего обновления.



Для каждой базы знаний доступно контекстное меню:

- **Документы** – открывает список документов базы знаний. Перейти к документам можно и по клику в любом места строки базы.
- **Настройки** – отображает параметры базы знаний, которые можно изменить -- название, настройки подключения к Confluence, а также эмбеддинга и чанкирования, которые доступны только для просмотра.
- **Удаление** – удаляет базу знаний со всеми документами и индексом.

Создание базы знаний

Необходимые условия:

- Документы подготовлены в одном из поддерживаемых форматов.
- Размер каждого файла не превышает 100 МБ. Может быть изменен в настройках сервиса shrag, отвечающего за функциональность.
- Есть лимит одновременной загрузки – не более 5 документов.

Поддерживаемые форматы:

- Текстовые документы – TXT, MD, CSV;
- Офисные документы – DOC, DOCX, XLS, XLSX, XLT, XLTX, PPTX, PPT;
- PDF-документы – PDF;
- Веб-документы – HTML, HTM.

ПРИМЕЧАНИЕ

Изображения и RAG:

- Изображения (JPG, GIF, PNG, WEBP) **не поддерживаются** для загрузки через базу знаний
- Вложения-картинки из Confluence также **не загружаются**
- Для работы с изображениями используйте блок **Загрузчик файлов** в связке с LLM-блоком, настроенными на работу с VLM (Vision-модели для распознавания содержимого)

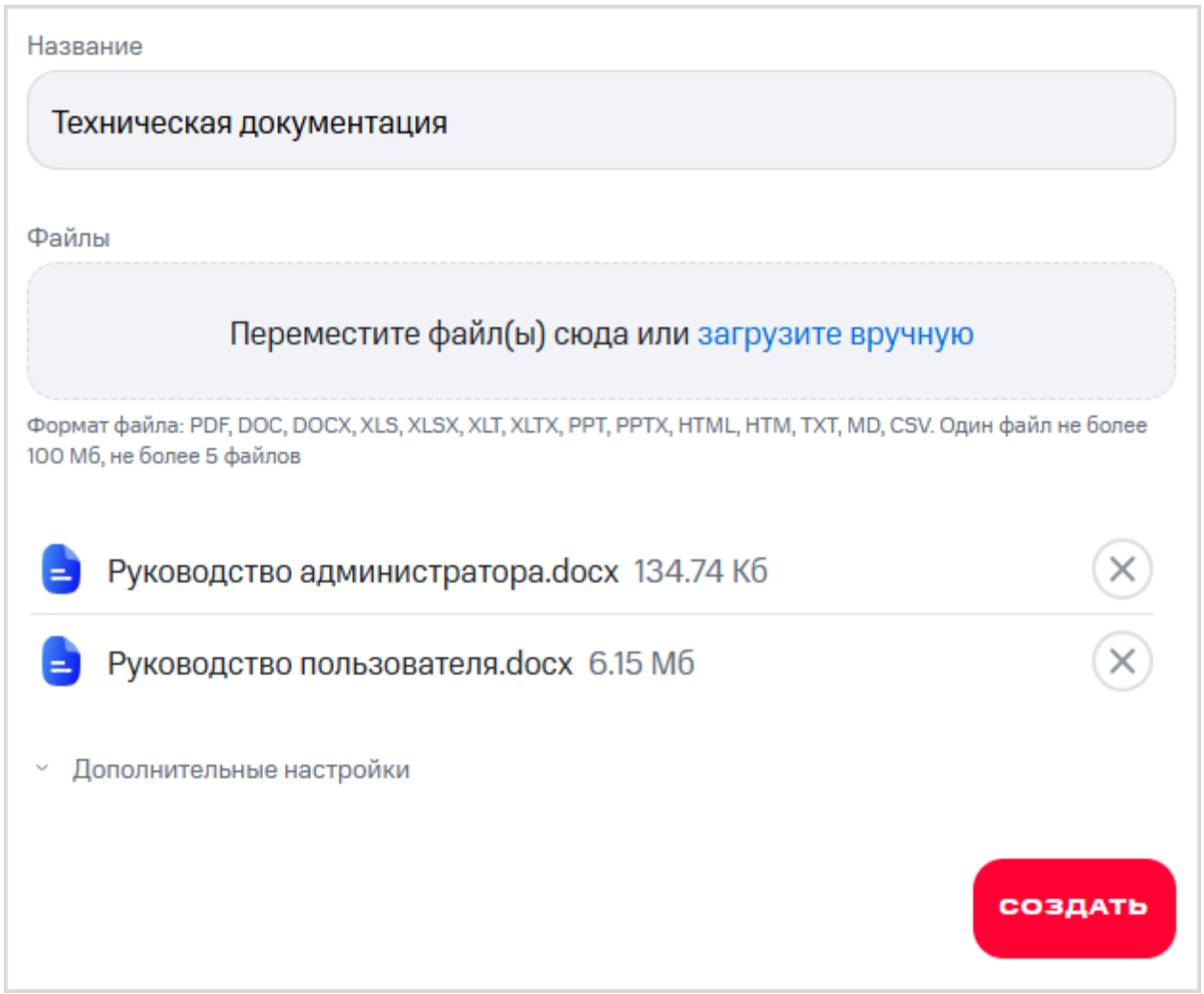
ПРИМЕЧАНИЕ

Особенности парсинга:

- **Таблицы** (в DOCX, HTML, PPTX, XLSX) преобразуются в строки формата «Key: Value» по каждой строке таблицы.
- **Excel** (XLSX, XLS) – каждая строка извлекается с заголовками колонок.
- **PDF** поддерживается в двух режимах: `text` (быстрое извлечение) и `markdown` (анализ layout, медленнее).

Чтобы создать базу знаний:

1. В разделе **Проекты** выберите проект. Перейдите в раздел **База знаний**.
2. Нажмите на кнопку **Создать базу знаний**. В поле **Название** введите понятное имя,например «FAQ по продукту» или «Техническая документация».
3. Перетащите документы в область **Файлы** или нажмите для выбора вручную. Кроме того, можно создать пустую базу знаний и загрузить документы позднее – просто пропустите этот шаг.
4. При необходимости разверните раздел **Дополнительные настройки** и укажите режимы поиска, параметры эмбеддинга и чанкинга. Подробнее см. в разделе «Дополнительные настройки».
5. Нажмите **Создать**.



После создания платформа автоматически начинает индексацию загруженных документов в фоновом режиме.

Добавление документов

Вы можете добавлять новые документы после создания БЗ.

1. В списке баз знаний найдите вашу базу.
2. Перейдите в раздел **Документы**.
3. Нажмите **Добавить файлы**.
4. Выберите или перетащите файлы в поле загрузки.

Добавленные документы проходят тот же цикл обработки:

- Ожидание → Загрузка → Загружен → Индексация → Проиндексирован

После индексации документ становится доступен для поиска.

> **ПРИМЕЧАНИЕ**
>
> Для базы знаний с источником Confluence добавление файлов невозможно — используйте синхронизацию.

Статусы документов

Каждый документ в базе знаний проходит следующие этапы обработки:

СТАТУС	ОПИСАНИЕ
Ожидание	Документ ожидает обработки
Загрузка	Файл загружается в хранилище
Загружен	Файл загружен, ожидает чанкинга и эмбеддинга
Индексация	Выполняется чанкинг и преобразование в векторы (эмбеддинг)
Проиндексирован	Документ индексирован и готов к поиску
Пустой	Документ пустой, не содержит текста (не участвует в поиске)
Ошибка	Произошла ошибка при обработке (см. сообщение об ошибке)

> **ПОДСКАЗКА**
>
> Для поиска доступны только документы в статусе «Проиндексирован». Документы в статусах «Ошибка» или «Пустой» не участвуют в поиске.
>
> Практические советы:
>
> - Чтобы выяснить причину ошибки, наведите курсор на статус – появится всплывающее окно с текстом ошибки
> - Индексация выполняется в фоновом режиме и **не блокирует поиск** по уже индексированным документам. Чанки документов, находящихся в процессе индексации, не будут включены в результаты поиска до завершения обработки.

Управление документами

Обновление документов

Чтобы обновить содержимое документа, используйте один из двух способов:

Способ 1: Удалить и загрузить заново

- Удалите старую версию документа из базы
- Загрузите новую версию

Способ 2: Перезагрузить с тем же именем

- Загрузите новую версию документа с тем же именем
- Файл автоматически перезапишется в базе знаний
- Документ пройдёт полный цикл обработки: парсинг → чанкирование → создание эмбеддингов

Удаление документов

Чтобы удалить документ из базы знаний:

1. Перейдите в раздел **Базы знаний** и откройте нужную базу.
2. Выберите документ в списке и нажмите **Удалить**.
3. Подтвердите удаление.

При удалении документа из базы он автоматически удаляется и из индекса.

Confluence как источник базы знаний

Помимо ручной загрузки файлов, база знаний может быть создана на основе данных из Confluence Server/Data Center.

Особенности базы знаний с Confluence

- Ручная загрузка файлов в такую базу **недоступна**
- Доступна синхронизаия с Confluence
- Каждый документ содержит ссылку на страницу в Confluence

Создание базы знаний с Confluence

Подробнее см. Confluence как источник базы знаний.

1. В разделе **Проекты** выберите проект. Перейдите в раздел **База знаний**.
2. Нажмите **Создать базу знаний**.
3. Выберите **Тип источника** -- **Confluence**.
4. Введите название базы, например «Confluence: Документация».

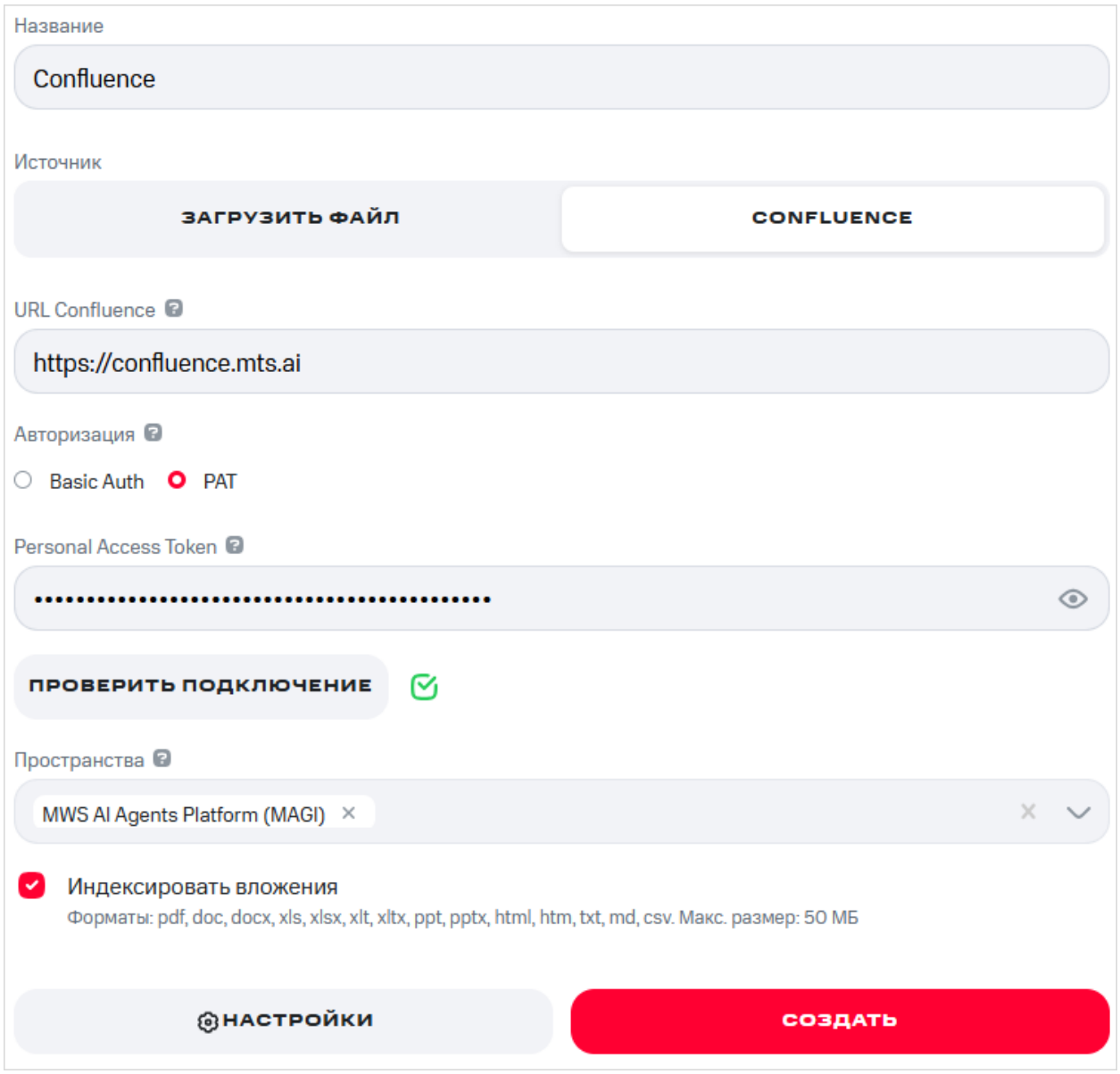
5. Заполните параметры:

ПАРАМЕТР	ОПИСАНИЕ
URL Confluence	Адрес Confluence, например `https://confluence.example.com`
Тип аутентификации	`BASIC` (логин/пароль) или `PAT` (Personal Access Token)
Имя пользователя	Имя пользователя (для BASIC) сервисного акаунта Confluence с правами чтения
Пароль	Пароль (для BASIC) сервисного акаунта Confluence
PAT	Personal Access Token (для PAT)

6. Когда все параметры внесены, активируется кнопка **Проверить подключение**. Нажмите ее, чтобы выпонить тестовое подключение и получит список пространств Confluence.
7. Если подключение успешно, появятся статус успеха  и поле **Пространства**
8. Выберите нужные ключи пространств для синхронизации.
9. Снимите флажок **Индексировать вложения**, если не нужно синхронизировать вложения страниц.

> **ПРИМЕЧАНИЕ**
>
> ```
> Вложения Confluence: Текстовые вложения (DOCX, XLSX, PDF и др.)
> синхронизируются и индексируются. Вложения-картинки (JPG, PNG, GIF, WEBP)
> **не загружаются** — даже если флажок установлен, изображения игнорируются.
> ```

10. При необходимости разверните раздел **Дополнительные настройки** и укажите режимы поиска, параметры эмбеддинга и чанкинга. Подробнее см. в разделе «Дополнительные настройки».
11. Нажмите **Создать**.



Запуск синхронизации

1. В списке баз знаний найдите вашу базу знаний Confluence.
2. Нажмите **Синхронизировать**.
3. Ожидайте завершения синхронизации (статус «Синхронизация» → «Готово»).

Статусы синхронизации Confluence

СТАТУС	ОПИСАНИЕ
Не активна	Синхронизация не активна
Синхронизация	Выполняется синхронизация

СТАТУС	ОПИСАНИЕ
Ошибка	Произошла ошибка (см. сообщение)
Готово	Синхронизация завершена успешно

ПОДСКАЗКА

Повторная синхронизация пропускает страницы, неизменённые с момента предыдущей синхронизации (выполняется проверка по хэшу содержимого). Документы в статусе «Ошибка» или «Пустой» переиндексируются автоматически.

Дополнительные настройки

Дополнительные настройки задаются при создании базы знаний и используются во всех операциях поиска. Изменить их после создания базы нельзя — если нужны другие параметры, создайте новую базу знаний.

Режимы поиска

Поиск по базе знаний может выполняться в двух режимах:

РЕЖИМ	ОПИСАНИЕ	КОГДА ИСПОЛЬЗОВАТЬ
Векторный (семантический) поиск	Поиск по смыслу. Вопрос и документы преобразуются в векторы, ищутся наиболее похожие по смыслу фрагменты	Когда важно найти релевантную по смыслу информацию, даже если слова в вопросе не совпадают с документом
Полнотекстовый поиск (BM25)	Поиск по ключевым словам. Ищутся фрагменты с точным совпадением или близостью слов	Когда важна точность совпадения терминов (например, поиск по названиям команд, параметров, специфичных терминов)

ПРИМЕЧАНИЕ

Режимы поиска задаются при создании базы знаний и не могут быть изменены. Можно выбрать один или оба режима одновременно. В последнем случае, становится доступен **Гибридный поиск**, который настраивается в блоке реакции **Индекс**.

Гибридный поиск:

Гибридный поиск комбинирует векторный и полнотекстовый поиск с настраиваемым балансом весов. Решает ряд проблем, которые не покрывает ни один из режимов отдельно:

- **Точные совпадения на редких токенах** — векторный поиск плохо различает похожие коды и номера (например, KZ-4421-B vs KZ-4422-B), а BM25 ловит точное совпадение;

- **Внутренний жаргон и специфичные термины** — векторная модель, обученная на общем корпусе, может не понимать продуктовые названия и сокращения, но BM25 найдёт по буквам;
- **Семантические парафразы** — векторный поиск найдёт релевантный документ, даже если пользователь задал вопрос другими словами;
- **Короткие запросы** — на 1–3 словах BM25 точнее воспринимает намерение пользователя, чем векторный поиск;
- **Многоязычие и опечатки** — BM25 устойчив к смеси языков и небольшим опечаткам;
- **Повышение recall** — объединение двух каналов даёт выше recall, чем каждый по отдельности.

Баланс гибридного поиска:

При использовании гибридного режима результаты двух каналов можно комбинировать в различных соотношениях веса:

- **Векторный поиск** — вес по умолчанию **0.70** (70%)
- **Полнотекстовый** — вес по умолчанию **0.30** (30%)

Баланс весов настраивается ползунком в блоке реакции **Индекс** и может быть отрегулирован под конкретную задачу.

Настройки эмбеддинга

Эмбеддинг – процесс преобразования текста в числовое представление (вектор), по которому выполняется семантический поиск.

Параметры:

- **URL модели** – адрес инференс-сервера для эмбеддинга. Оставьте пусто для использования системного эмбедера (по умолчанию `Triton`);
- **API-ключ** – ключ для доступа к сервису эмбеддинга (требуется только для сторонних моделей);
- **Модель** – имя модели эмбеддинга. Выберите из доступного списка или укажите вручную;
- **Размер батча** – количество фрагментов, обрабатываемых за один запрос. По умолчанию `50`. Увеличение ускоряет эмбеддинг, но требует больше памяти.

> **ПОДСКАЗКА**
>
> Для использования системного эмбедера оставьте поля **URL модели** и **API-ключ** пустыми. Значения будут установлены автоматически.

Настройки чанкирования

Чанкирование – разбиение текста документа на фрагменты (чанки), по которым выполняется поиск. Эти параметры постоянны для базы и применяются ко всем документам.

Параметры:

- **Тип чанкирования** – алгоритм разбиения:
  - **По токенам** (по умолчанию в сценариях) — разбиение по количеству токенов. Точный контроль размера чанка;

    - **По предложениям** — разбиение по границам предложений. Сохраняет целостность предложений;
    - **Рекурсивный** — сначала разбивает по абзацам, потом по предложениям. Подходит для структурированных документов (книги, техдокументация);
- **Размер чанка** – максимальный размер одного фрагмента (в токенах). По умолчанию `350`. Влияет на точность поиска и размер контекста;
- **Перекрытие** – количество токенов, общих для соседних фрагментов. По умолчанию `70`. Помогает сохранить контекст между чанками.

Создание RAG со статическим индексом

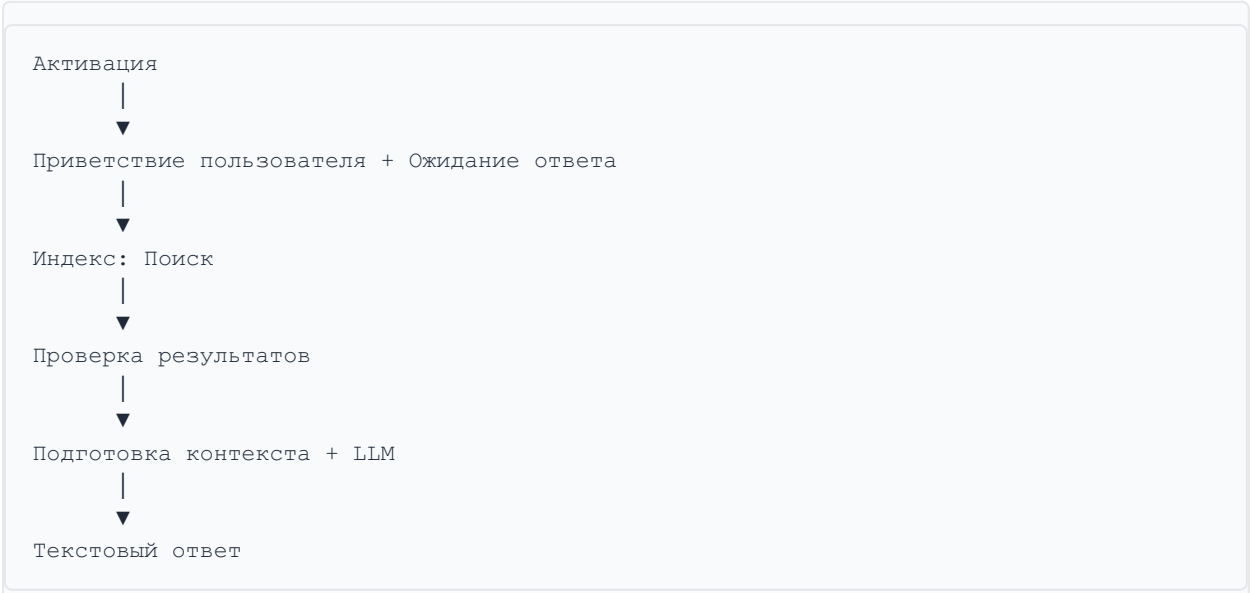
В разделе описывается RAG-конвейер, который отвечает на вопросы пользователя по заранее подготовленной базе знаний.

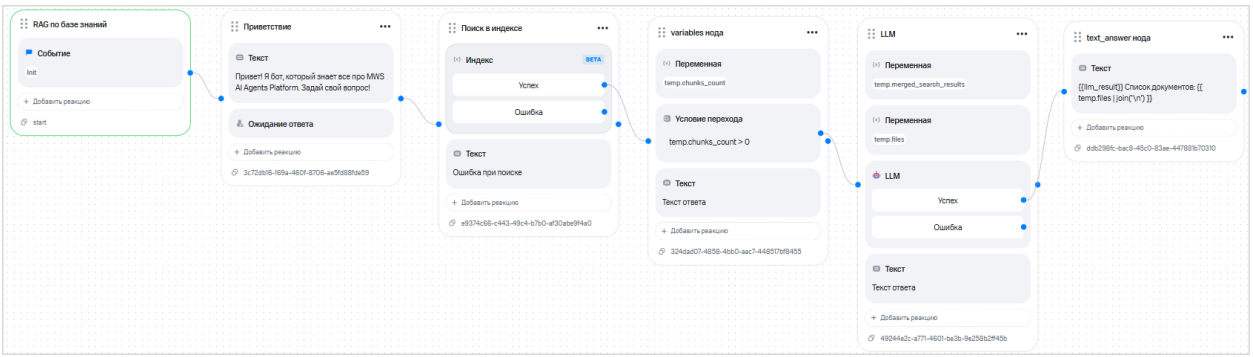
- Бот отвечает по заранее известным документам: FAQ, регламенты, инструкции, описания продуктов.
- Одна база знаний обслуживает всех пользователей.
- Документы обновляются редко – через интерфейс платформы.

Необходимые условия

- Проект с созданной версией.
- База знаний с загруженными документами. Все документы должны быть проиндексированы. Подробнее о создании базы знаний см. раздел «Базы знаний».

Базовая логика сценария





Пошаговая инструкция

> **ПОДСКАЗКА**
>
> Подробнее о работе с конструктором — в разделе «Работа в конструкторе». Информация о блоках сценария собрана в разделе «Компоненты сценария».

Шаг 1. Создайте сценарий и настройте активацию

В конструкторе сценариев создайте новый сценарий — например, «RAG по базе знаний». В стартовую ноду добавьте блок активации нужного типа в зависимости от вашей задачи.

Шаг 2. Добавьте приветствие и ожидание вопроса

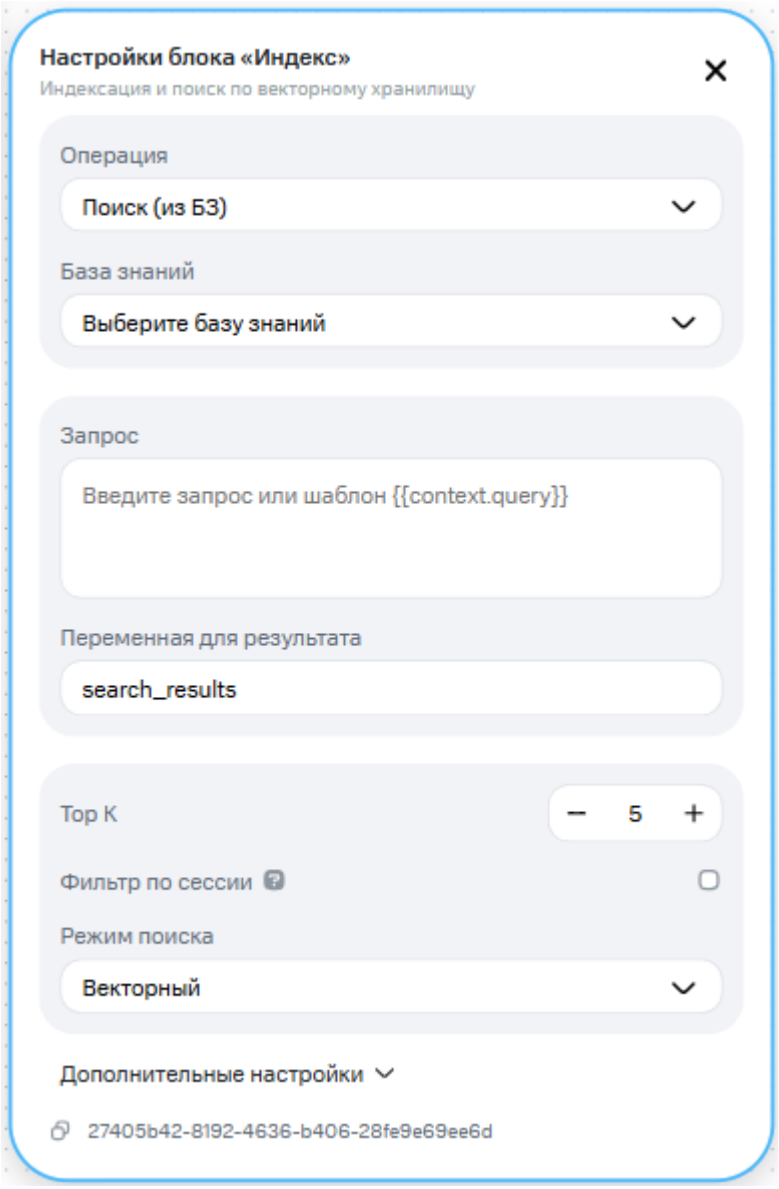
1. Перенесите на рабочее поле конструктора блок **Текст**.
2. Добавьте сообщение-приглашение для пользователя. Например, «Привет! Я бот, который знает все про MWS AI Agents Platform. Задай свой вопрос!»
3. В ноду блока **Текст** добавьте блок **Ожидание ответа**, чтобы приостановить выполнение сценария до получения сообщения от пользователя.

> **ПРИМЕЧАНИЕ**
>
> Шаг 2 не обязателен. Если сценарий активируется по **No Match**, вопрос пользователя уже содержится в зарезервированной переменной {{system.source_message}}, и вы можете сразу переходить к поиску. Подробнее о переменных см.раздел «Зарезервированные переменные».

Шаг 3. Настройте индекс для поиска

1. Перенесите на рабочее поле конструктора блок **Индекс**
2. Задайте следующие настройки:

ПАРАМЕТР	ЗНАЧЕНИЕ
Операция	Поиск (из БЗ)
База знаний	Выберите нужную базу знаний
Запрос	`{{system.source_message}}`
Top K	5 – 10
Режим поиска	Гибридный (по умолчанию) — комбинирует векторный (вес 0.70) и BM25 поиск (вес 0.30). Баланс настраивается слайдером. Можно выбрать отдельно Векторный или Полнотекстовый.
Фильтр по сессии	Выключен (для статического индекса используется общая база для всех пользователей)
Переменная для результата	temp.search_results
Переход при успехе	Проверка результата



3. В эту же ноду добавьте блок **Текстовый ответ** с сообщением об ошибке, например «Ошибка при поиске». Он сработает только в случае сбоя блока **Индекс**.

Шаг 4. Добавьте проверку результатов

Поиск может не найти подходящих фрагментов в базе знаний. Чтобы бот не отправлял запрос в LLM с пустым контекстом, добавьте проверку на наличие в контексте чанков из индекса.

1. Добавьте в сценарий блок **Переменная** для подсчёта количества результатов чанкирования:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.chunks_count
Тип	Python
Значение	len(temp.search_results)

2. Добавьте в ноду блок **Условие перехода** – количество чанков больше нуля:

ПАРАМЕТР	ЗНАЧЕНИЕ
Выражение	temp.chunks_count > 0
Нода для перехода	Следующий этап сценария – **LLM**

3. Добавьте ноду блок **Текст** с ответом, который будет выдан пользователю, если условие в предыдущем блоке не выполнится: «По вашему вопросу ничего не найдено. Попробуйте переформулировать.»

Шаг 5. Подготовьте контекст и добавьте LLM

Поместите в сценарий блок LLM и к нему в ноду добавьте блоки в следующем порядке:

1. Блок **Переменная** – для склейки текстов найденных фрагментов:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.merged_search_results
Тип	Python
Значение	"\n\n".join([r.get("text", r) for r in temp.search_results])

2. Блок **Переменная** – для извлечение имён файлов-источников:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.files
Тип	Python
Значение	set(r.get("file_name", r) for r in temp.search_results)

3. Блок **LLM** – для генерации ответа.
4. Настройте параметры **LLM**:
   - на вкладке **Промпт** в поле **Системное сообщение** укажите инструкцию для модели.

     Пример:

```
Ты полезный ассистент. Используй ТОЛЬКО информацию из предоставленного
КОНТЕКСТА для ответа на ВОПРОС.

Пример пользовательского промпта:
---
КОНТЕКСТ: "текст"

ВОПРОС: "текст"
---
```

```
ИНСТРУКЦИЯ: Ответь на вопрос пользователя, используя представленный выше
контекст, БЕЗ упоминания контекста. Ответ должен быть основан на фактах из
контекста. Ответь кратко, естественно, без упоминания КОНТЕКСТА или шагов.

ШАГ 1: Определи ключевые факты из КОНТЕКСТА, релевантные ВОПРОСУ (перечисли 2-3
bullet'a).
ШАГ 2: Если КОНТЕКСТ содержит хоть какую-то релевантную информацию — дай полный
обоснованный ответ на основе неё. Даже если ответ частичный.
ШАГ 3: Только если КОНТЕКСТ полностью не касается ВОПРОСА (0% overlap) —
ответь: "Недостаточно информации".
```

- в поле **Пользовательское сообщение** подставьте контекст и вопрос.

  Пример:

  ```
  КОНТЕКСТ: '{{ temp.merged_search_results }}'.
  ВОПРОС: '{{system.last_user_message}}'.Где:
  temp.merged_search_results - склеенные тексты найденных фрагментов;
  system.last_user_message - последнее сообщение пользователя.
  ```

- на вкладке LLM задайте параметры модели. Описание параметров см. в разделе «LLM».
- на вкладке **Результат** в поле **Переменная** введите имя переменной для сохранения результата, например, llm_result.
- укажите в качестве перехода при успехе следующую ноду.

5. В эту же ноду добавьте блок **Текст** с сообщением «Ошибка». Он сработает при сбое LLM.

Шаг 6. Подготовьте вывод ответа пользователю

Добавьте в сценарий блок **Текст** с переменными, из которых будет собираться ответ.

Пример:

```
{{llm_result}}
Список документов: {{ temp.files | join('\n') }}

Где:

llm_result - ответ, сгенерированный языковой моделью;

temp.files - набор имён файлов, из которых были найдены фрагменты.
```

Сценарий собран. Можно выполнить отладку и тестирование. Подробнее см. в разделе «Запуск проекта».

Альтернативные варианты

- Создание RAG с динамическим индексом — если пользователи загружают свои документы

Карта переменных сценария

```
Индекс: Поиск (результат)  →  temp.search_results
                                      ↓
Переменная (подготовка)     →  temp.merged_search_results
                            →  temp.files
                                      ↓
LLM (промпт)               ←  temp.merged_search_results
LLM (результат)             →  llm_result
                                      ↓
Текстовый ответ             ←  llm_result
```

Создание RAG с динамическим индексом

В разделе описывается как создать RAG-пайплайн, в котором клиент загружает документ в процессе диалога с ботом и задаёт по нему вопросы. Сценарий подойдет в ситуациях:

- клиент хочет получить ответы по содержанию своего документа;
- каждый пользователь бота работает со своими файлами. Данные должны быть изолированы;
- не требуется хранить документы после завершения диалога.

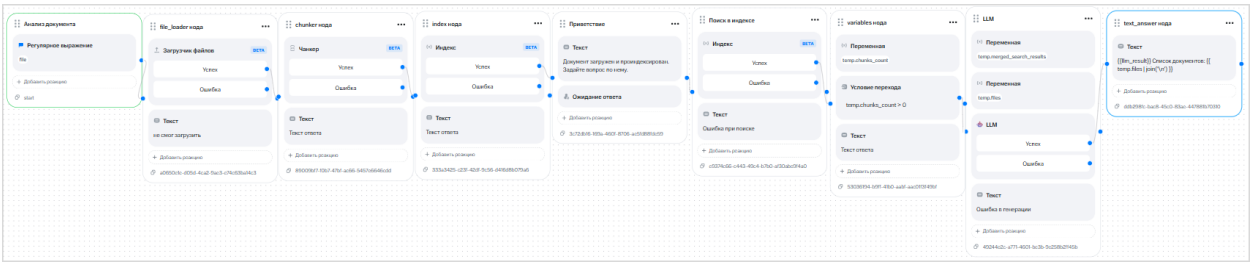
Необходимые условия

- Проект с созданной версией.
- База знаний с загруженными документами. Документы в ней не обязательны – блок **Индекс** в режиме загрузки чанков использует только параметры эмбеддера из базы знаний. О создании базы знаний см. раздел «Базы знаний».

Базовая логика сценария

```
Активация
      |
      ▼
Загрузчик файлов
      |
      ▼
Чанкер
      |
      ▼
Индекс: Загрузка чанков
      |
      ▼
Подтверждение + Ожидание вопроса
      |
      ▼
Индекс: Поиск
```

```
        |
        ▼
Подготовка контекста + LLM
        |
        ▼
Текстовый ответ
```



Пошаговая инструкция

> **ПОДСКАЗКА**
>
> Подробнее о работе с конструктором — в разделе «Работа в конструкторе». Информация о блоках сценария собрана в разделе «Компоненты сценария».

Шаг 1. Создайте сценарий и настройте активацию

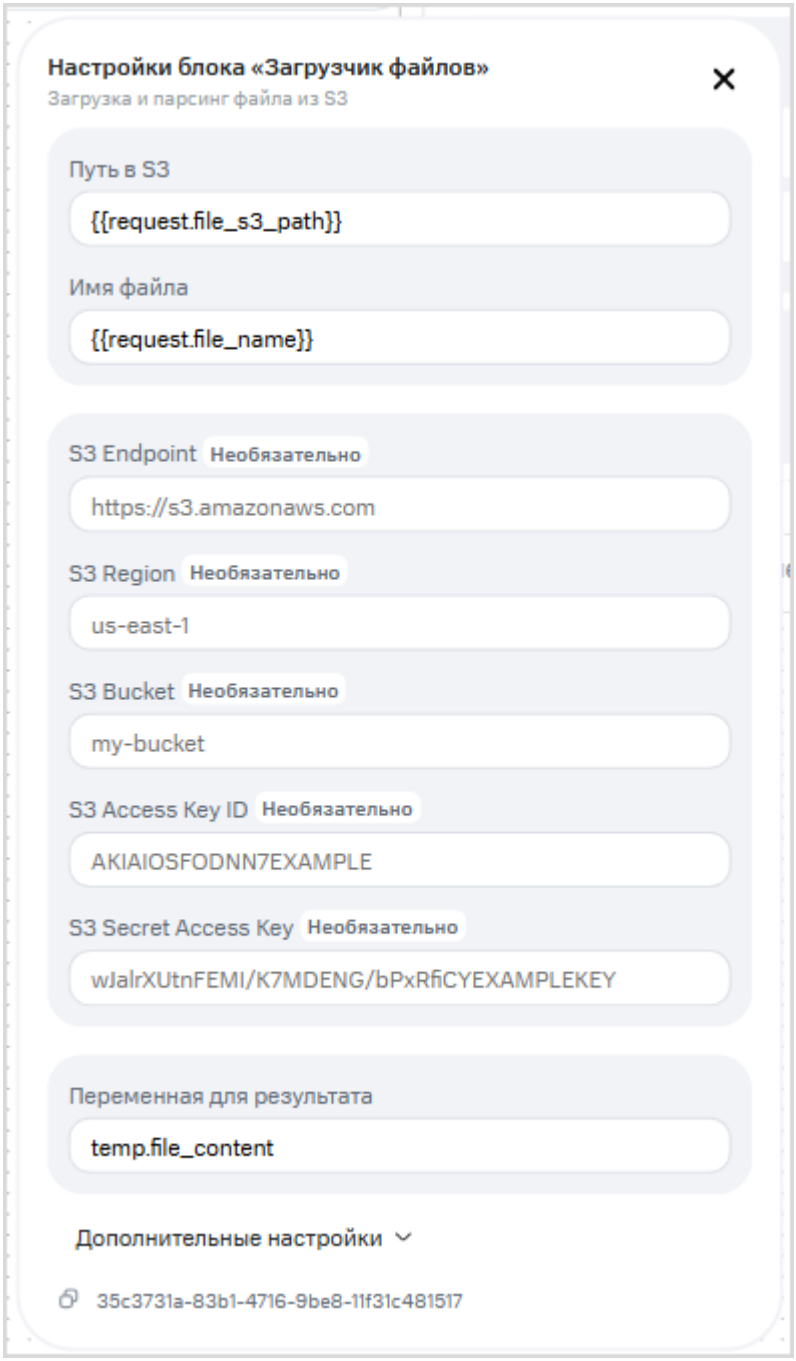
В конструкторе сценариев создайте новый сценарий — например «Анализ документа».

В стартовую ноду добавьте блок активации с типом **Регулярное выражение** — например, **file**. Сценарий запустится, когда пользователь отправит файл.

Шаг 2. Добавьте блок «Загрузчик файлов»

1. Перенесите в рабочую область блок **Загрузчик файлов** и настройте параметры.

ПАРАМЕТР	ЗНАЧЕНИЕ
Путь в S3	{{request.file_s3_path}}
Имя файла	{{request.file_name}}
Переменная для результата	temp.file_content
Переход при успехе	Следующая нода (Чанкер)



2. В эту же ноду добавьте блок текст с ответом при ошибке загрузки.

Шаг 3. Добавьте блок «Чанкер»

1. Перенесите на рабочее поле конструктора следующий блок – **Чанкер**. Настройте его параметры.

ПАРАМЕТР	ЗНАЧЕНИЕ
Переменная с содержимым	temp.file_content
Тип разбиения	По токенам (доступны: По токенам, По предложениям, Рекурсивный)
Размер чанка	512

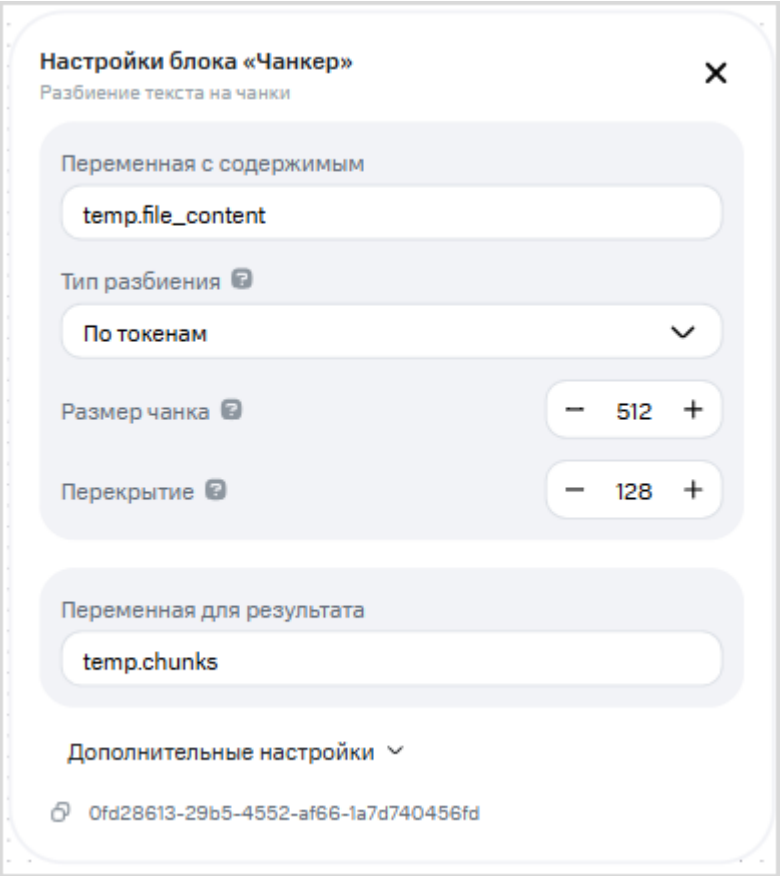
ПАРАМЕТР	ЗНАЧЕНИЕ
Перекрытие	128
Переменная для результата	temp.chunks
Переход при успехе	Следующая нода (Индекс: Загрузка чанков)

2. В эту же ноду добавьте блок **Текст** с ответом при ошибке чанкирования.

Шаг 4. Добавьте блок Индекс для загрузки чанков

1. Добавьте в сценарий блок **Индекс** со следующими настройками:

ПАРАМЕТР	ЗНАЧЕНИЕ
Операция	Загрузка чанков
База знаний	Выберите базу знаний с подходящими параметрами эмбеддера
Переменная с чанками	temp.chunks
Переменная для результата	temp.ingest_result
Переход при успехе	Следующая нода (Подтверждение и ожидание загрузки)



2. Добавьте в ноду блок **Текст** с ответом пользователю, если при индексации возникли ошибки.

Шаг 5. Подготовьте подтверждение загрузки и ожидание вопроса

1. Добавьте блок **Текстовый ответ**. Введите текст подтверждения, например: «Документ загружен и проиндексирован. Задайте вопрос по нему.»
2. Добавьте блок **Ожидание сообщения** в ноду к блоку **Текстовый ответ**.

> **ПОДСКАЗКА**
>
> Этот шаг разделяет сценарий на две фазы: **индексацию** (действия до текущего шага) и **вопрос-ответ** (действия после текущего шага). Пользователь видит подтверждение и понимает, что можно задавать вопросы.

Шаг 6. Добавьте блок Индекс для поиска

1. Добавьте в сценарий второй блок **Индекс**. Настройте его параметры:

ПАРАМЕТР	ЗНАЧЕНИЕ
Операция	Поиск (search_kb)
База знаний	Та же база знаний, что и в шаге 4
Запрос	`{{system.source_message}}`
Top K	5
Режим поиска	Гибридный (по умолчанию) — комбинирует векторный (вес 0.70) и BM25 поиск (вес 0.30). Баланс настраивается слайдером. Можно выбрать отдельно Векторный или Полнотекстовый.
Фильтр по сессии	Включён
Переменная для результата	temp.search_results
Переход при успехе	Следующая нода (подготовка контекста)

2. Добавьте в ноду блок **Текст** с ответом пользователю, если при поиске возникли ошибки.

> **ВНИМАНИЕ**
>
> Фильтр по сессии должен быть включён. Он ограничивает поиск фрагментами, загруженными в текущей сессии, чтобы документы одного пользователя не пересекались с документами другого. Это критично для динамического индекса, где каждый пользователь работает с собственными файлами.
>
> Оба блока **Индекс** (загрузка чанков и поиск) должны указывать на одну и ту же базу знаний, так как параметры эмбеддера и чанкирования жёстко привязаны к базе.

Шаг 7. Добавьте дополнительную проверку результатов

1. Добавьте проверку на наличие в контексте чанков из индекса, чтобы бот не отправлял запрос в LLM с пустым контекстом. Это нужно на случай, если при поиске в базе знаний не нашлись подходящие фрагменты.
2. Добавьте в сценарий блок **Переменная** для подсчёта количества результатов чанкирования:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.chunks_count
Тип	Python
Значение	len(temp.search_results)

3. Добавьте в ноду блок **Условие перехода** для количества чанков больше нуля:

ПАРАМЕТР	ЗНАЧЕНИЕ
Выражение	temp.chunks_count > 0
Нода для перехода	Следующие блок сценария – **LLM**

4. Добавьте в ноду блок **Текст** с ответом, который будет выдан пользователю, если условие в предыдущем блоке не выполнится, например: «По вашему вопросу ничего не найдено. Попробуйте переформулировать.»

Шаг 8. Подготовьте контекст и добавьте LLM

В сценарий добавьте блок **LLM** и в его ноду добавьте блоки в следующем порядке:

1. Блок **Переменная** для склейки текстов найденных фрагментов:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.merged_search_results
Тип	Python
Значение	"\n\n".join([r.get("text", r) for r in temp.search_results])

2. Блок **Переменная** для извлечения имён файлов-источников:

ПАРАМЕТР	ЗНАЧЕНИЕ
Имя переменной	temp.files
Тип	Python
Значение	set(r.get("file_name", r) for r in temp.search_results)

3. Блок **LLM** – для генерации ответа.
4. Настройте параметры **LLM**:
   - на вкладке **Промпт** в поле **Системное сообщение** укажите инструкцию для модели.

     Пример:

     ```
     Ты полезный ассистент. Используй ТОЛЬКО информацию из предоставленного
     КОНТЕКСТА для ответа на ВОПРОС.

     Пример пользовательского промпта:
     ---
     КОНТЕКСТ: "текст"

     ВОПРОС: "текст"
     ---

     ИНСТРУКЦИЯ: Ответь на вопрос пользователя, используя представленный выше
     контекст, БЕЗ упоминания контекста. Ответ должен быть основан на фактах из
     контекста. Ответь кратко, естественно, без упоминания КОНТЕКСТА или шагов.

     ШАГ 1: Определи ключевые факты из КОНТЕКСТА, релевантные ВОПРОСУ (перечисли 2-3
     bullet'a).
     ШАГ 2: Если КОНТЕКСТ содержит хоть какую-то релевантную информацию — дай полный
     обоснованный ответ на основе неё. Даже если ответ частичный.
     ШАГ 3: Только если КОНТЕКСТ полностью не касается ВОПРОСА (0% overlap) —
     ответь: "Недостаточно информации".
     ```

   - в поле **Пользовательское сообщение** подставьте контекст и вопрос.

     Пример:

     ```
     КОНТЕКСТ: '{{ temp.merged_search_results }}'.
     ВОПРОС: '{{system.last_user_message}}'.Где:
     temp.merged_search_results - склеенные тексты найденных фрагментов;
     system.last_user_message - последнее сообщение пользователя.
     ```

   - на вкладке **LLM** задайте параметры модели. Описание параметров см. в разделе «LLM».
   - на вкладке **Контекст** в поле **Переменная** введите имя переменной для сохранения результата, например, llm_result.
   - укажите следующую ноду для перехода при успехе.
5. В эту же ноду добавьте блок **Текст** с сообщением «Ошибка». Он сработает при сбое LLM.

Шаг 9. Подготовьте вывод ответа пользователю

Добавьте в сценарий блок **Текст** с переменными, из которых будет собираться ответ.

Пример:

```
{{llm_result}}
Список документов: {{ temp.files | join('\n') }}
```

```
Где:
llm_result - ответ, сгенерированный языковой моделью;
temp.files - набор имён файлов, из которых были найдены фрагменты.
```

Сценарий собран. Можно выполнить отладку и тестирование. Подробнее см. в разделе «Запуск и использование».

Сравнение с другими подходами

- Создание RAG со статическим индексом — когда БЗ единая для всех пользователей

Карта переменных сценария

```
Загрузчик файлов (результат) →  temp.file_content
                                      ↓
Чанкер (результат)            →  temp.chunks
                                      ↓
Индекс: Загрузка (чанки)      ←  temp.chunks
Индекс: Загрузка (результат)  →  temp.ingest_result
                                      ...
Индекс: Поиск (результат)     →  temp.search_results
                                      ↓
Переменная (подготовка)       →  temp.merged_search_results
                              →  temp.files
                                      ↓
LLM (промпт)                  ←  temp.merged_search_results
LLM (результат)               →  temp.llm_result
                                      ↓
Текстовый ответ               ←  temp.llm_result
```

Интеграция RAG-пайплайнов в сценарии бота

RAG-пайплайн – это отдельный сценарий. Бот может содержать несколько сценариев: приветствие, FAQ, обработка заявок, поиск по базе знаний и другие. Платформа выбирает нужный сценарий на основе активационных блоков.

Типовые схемы активации

ЗАДАЧА	ТИП АКТИВАЦИИ RAG-СЦЕНАРИЯ	ОПИСАНИЕ
RAG как основной режим	No Match	Если ни один другой сценарий не подошёл – запрос уходит в RAG. Бот сначала пробует точные сценарии, затем «ловит» всё остальное через RAG

ЗАДАЧА	ТИП АКТИВАЦИИ RAG-СЦЕНАРИЯ	ОПИСАНИЕ
RAG по команде	Регулярное выражение	Сценарий срабатывает по ключевому слову или команде – например, **/ask** или найди в документации
RAG по событию с поверхности	Custom Event	Сценарий запускается внешним событием – например, нажатием кнопки в интерфейсе
Динамический RAG при загрузке файла	Регулярное выражение (**file**)	Сценарий срабатывает, когда пользователь отправляет файл

Пример. Бот с двумя типами RAG

Бот содержит три сценария:

```
Сценарий 1: «Приветствие» Активация: Custom Event (start)
Сценарий 2: «RAG по базе знаний» (статический) Активация: No Match
Сценарий 3: «RAG по документу пользователя» (динамический) Активация: Регулярное
выражение (file)
```

Логика работы:

1. Пользователь начинает диалог – срабатывает сценарий «Приветствие».
2. Пользователь задаёт текстовый вопрос, платформа не находит точного совпадения – срабатывает **No Match**. Вопрос переходит в RAG со статическим индексом.
3. Пользователь отправляет файл, срабатывает переменная **file** – запускается RAG с динамическим индексом.

Совмещение RAG с классификатором

Если к проекту привязан классификатор, платформа сначала определяет тематику сообщения и направляет его в соответствующий сценарий. RAG-сценарий с активацией **No Match** срабатывает только для сообщений, которые классификатор не отнёс ни к одной из известных тематик.

Это позволяет строить гибкую маршрутизацию: точные вопросы обрабатываются специализированными сценариями, а RAG работает как резервный сценарий для всего остального.

Подробнее о классификаторах в проектах ботов см. раздел «Привязка классификатора».

Варианты реализации RAG

- Создание RAG со статическим индексом — для единой БЗ
- Создание RAG с динамическим индексом — для документов пользователя
- Расширение RAG-пайплайна — продвинутые техники

Расширение RAG-пайплайна

Базовый RAG-пайплайн можно расширить дополнительными блоками реакции. Ниже – типовые приёмы.

Постобработка результатов поиска

Добавьте блок **Скрипт** между блоком **Индекс** и блоком **LLM**, чтобы отфильтровать или ранжировать найденные фрагменты перед передачей в модель.

Например:

- Оставить только фрагменты с определённым именем файла.
- Отсечь фрагменты с низкой релевантностью (если поле **score** доступно).
- Ограничить общий объём контекста, чтобы не превысить окно модели.

HTTP-запрос перед LLM

Добавьте блок **HTTP-запрос** перед блоком **LLM**, чтобы обогатить контекст данными из внешней системы – например, получить актуальный статус заказа или курс валюты. Результат запроса сохраните в переменную и включите в промпт наряду с фрагментами из базы знаний.

Несколько баз знаний

Если бот работает с документами из разных доменов (например, техническая документация и юридические регламенты), добавьте несколько блоков «Индекс» на холст — каждый со своей базой знаний. Результаты поиска из разных баз объедините в блоке **Переменная** перед передачей в LLM.

Варианты базовой конфигурации

- Создание RAG со статическим индексом — стартовая точка
- Создание RAG с динамическим индексом — для документов пользователя
- Интеграция RAG-пайплайнов в сценарии — маршрутизация запросов

Рекомендации

Для создания и управления базами знаний

РЕКОМЕНДАЦИЯ	ПОЯСНЕНИЕ
Выбирайте источник данных	Для статических документов (FAQ, регламенты) загружайте файлы вручную. Для часто обновляемой документации (регламенты, инструкции) используйте Confluence — синхронизация будет автоматической.

РЕКОМЕНДАЦИЯ	ПОЯСНЕНИЕ
Планируйте параметры чанкирования заранее	Параметры нельзя изменить после создания базы. Если нужны другие параметры — создайте новую базу. Подбирайте размер чанка под тип документа: FAQ – 350–500 токенов, руководства – 500–700, статьи – 700–1000.
Выбирайте режим поиска под задачу	Для большинства случаев используйте режим **Гибридный** (комбинирует точность ключевых слов и семантический поиск). Используйте режим **Векторный**, если нужна релевантность по смыслу. Используйте режим **BM25**, если документы содержат специфичные термины и команды. Дефолтные веса гибридного режима — `0.70` для векторного, `0.30` для BM25.
Проверяйте статусы документов	Документы в статусе «Проиндексирован» готовы к поиску. Документы в статусах «Ошибка» или «Пустой» не участвуют в поиске — проверьте содержимое и переиндексируйте.
Мониторьте добавление документов	Для базы знаний с файлами вы можете добавлять документы после создания БЗ. Следите за статусом индексации — только документы в статусе «Проиндексирован» участвуют в поиске. Для типа Confluence используйте синхронизацию.
Мониторьте размер батча эмбеддера	По умолчанию **50**. Если индексация медленная, уменьшите батч. Если нужна скорость и у вас есть ресурсы — увеличьте батч.
Используйте собственный эмбедер, если нужны специфичные модели	Если системный эмбедер не подходит, укажите URL модели и API-ключ при создании базы. Убедитесь, что модель доступна из сети платформы.

Для разработчиков сценариев (параметры блока Индекс)

РЕКОМЕНДАЦИЯ	ПОЯСНЕНИЕ
Проверяйте пустые результаты поиска	Всегда добавляйте проверку **len(search_results) > 0** перед LLM. Без контекста модель будет генерировать ответы из общих знаний.
Ограничивайте Top K	Оптимальное значение — 5–10. Большие значения увеличивают контекст и могут снизить точность ответа.
Используйте гибридный режим с правильным балансом	По умолчанию баланс `0.70` векторный / `0.30` BM25 оптимален для большинства смешанных баз знаний (FAQ + техдокументация). Увеличивайте вес векторного поиска (> `0.70`) для семантически сложных текстов и описаний. Уменьшайте вес векторного поиска (< `0.70`) если документы содержат много специфичных терминов, кодов, названий команд (техдокументация, API-справочники). На ползунке в UI выберите нужное значение в диапазоне `0.0` – `1.0`.
Указывайте в промпте ограничение контекста	Формулировка «отвечай ТОЛЬКО на основе контекста» снижает вероятность галлюцинаций модели.

РЕКОМЕНДАЦИЯ	ПОЯСНЕНИЕ
Добавляйте fallback-ответ	Если поиск не нашёл фрагментов или они не релевантны, выводите понятное сообщение вместо пустого ответа.
Используйте скоупы переменных правильно	Данные текущего запроса — в **request**. Данные сценария — в **temp**. Данные проекта — в **project**.
Для динамического RAG включайте фильтр по сессии	В блоке Индекс (режим поиска) включайте **Фильтр по сессии**, чтобы документы одного пользователя не пересекались с документами другого. Подробнее см. Индекс блок.

Confluence как источник базы знаний

Помимо ручной загрузки файлов, база знаний может быть создана с источником данных из Confluence. Это позволяет автоматически синхронизировать документы из Confluence, поддерживать их в актуальном состоянии и использовать ссылки на исходные страницы в ответах.

Пошаговое создание БЗ с Confluence

Полное руководство по созданию, настройке и запуску синхронизации см. в разделе «Базы знаний» → Confluence как источник.

Когда использовать Confluence как источник

СЦЕНАРИЙ	РЕКОМЕНДАЦИЯ
Документация хранится в Confluence	Используйте синхронизацию
Документы часто обновляются	Автоматическое обновление при синхронизации
Требуется связывать ответы с оригинальными страницами	Каждый документ содержит ссылку `source_url`
Требуется ручная загрузка произвольных файлов	Используйте обычную базу знаний с `source_type=upload`

Особенности БЗ с Confluence

- Ручная загрузка файлов в такую базу **недоступна**
- Документы синхронизируются автоматически из Confluence
- Каждый документ содержит ссылку `source_url` на страницу в Confluence
- При повторной синхронизации неизменённые страницы пропускаются (проверка по хэшу содержимого)

Типы аутентификации

BASIC (логин/пароль)

Требует учётные данные сервисного акаунта Confluence с правами на чтение указанных пространств:

```
Username: имя_пользователя
Password: пароль
```

PAT (Personal Access Token)

Более безопасный способ аутентификации. Создайте PAT в Confluence и используйте его вместо пароля:

```
PAT: ATATT... (Personal Access Token)
```

PAT имеет ограниченный срок действия и может быть отозван без изменения пароля пользователя.

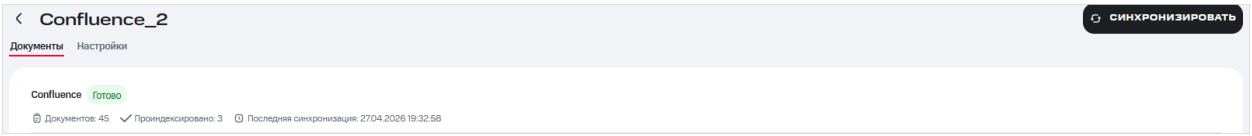
Статусы синхронизации Confluence

СТАТУС	ОПИСАНИЕ
Не активна	Синхронизация не активна
Синхронизация	Выполняется синхронизация
Ошибка	Произошла ошибка (см. сообщение об ошибке)
Готово	Синхронизация завершена успешно

Мониторинг синхронизации

Для отслеживания прогресса синхронизации:

1. Откройте базу знаний → **Статус синхронизации**.
2. Просмотрите счётчики документов:
    - **Всего** – общее количество документов
    - **Проиндексировано** – успешно обработано
    - **Последняя синхронизация** - её дата и время



Повторная синхронизация

При повторном запуске синхронизации:

- Неизменённые с даты последней синхронизации страницы пропускаются (проверка по хэшу содержимого)
- Новые страницы добавляются
- Обновлённые страницы переиндексируются
- Документы в статусе «Ошибка» или «Пустой» переиндексируются автоматически

> **ПОДСКАЗКА**
>
> Для полной пересинхронизации всех документов:
>
> 1. Удалить базу знаний
> 2. Создать новую с теми же параметрами
> 3. Запустить синхронизацию

Поиск по БЗ Confluence

При поиске по базе знаний с Confluence:

- Результаты содержат поле `source_url` – прямую ссылку на страницу в Confluence
- Можно указать источник в ответе пользователю

Пример использования в сценарии:

```
{{llm_result}}
Источник: {{ temp.source_urls | join('\n') }}
```

Ограничения

ОГРАНИЧЕНИЕ	ЗНАЧЕНИЕ
Макс. размер вложения	50 МБ
Количество пространств	Не ограничено

Troubleshooting

Проверка подключения не прошла

Убедитесь, что:

- URL Confluence доступен из сети платформы и в формате `https://confluence.example.com`
- Учётные данные верны (логин, пароль или PAT)
- У пользователя есть права на чтение указанных пространств в Confluence

Синхронизация зависает или долго выполняется

- Проверьте размер синхронизируемых пространств (большое количество страниц требует времени)
- Убедитесь, что сеть стабильна
- Попробуйте синхронизировать меньше пространств

Документы не индексируются

- Проверьте, что документы в статусе «Проиндексирован» (не «Пустой» и не «Ошибка»)
- Убедитесь, что страницы в Confluence содержат текстовое содержимое (не только изображения)
- Проверьте логи синхронизации на предмет сообщений об ошибках

Более подробная информация

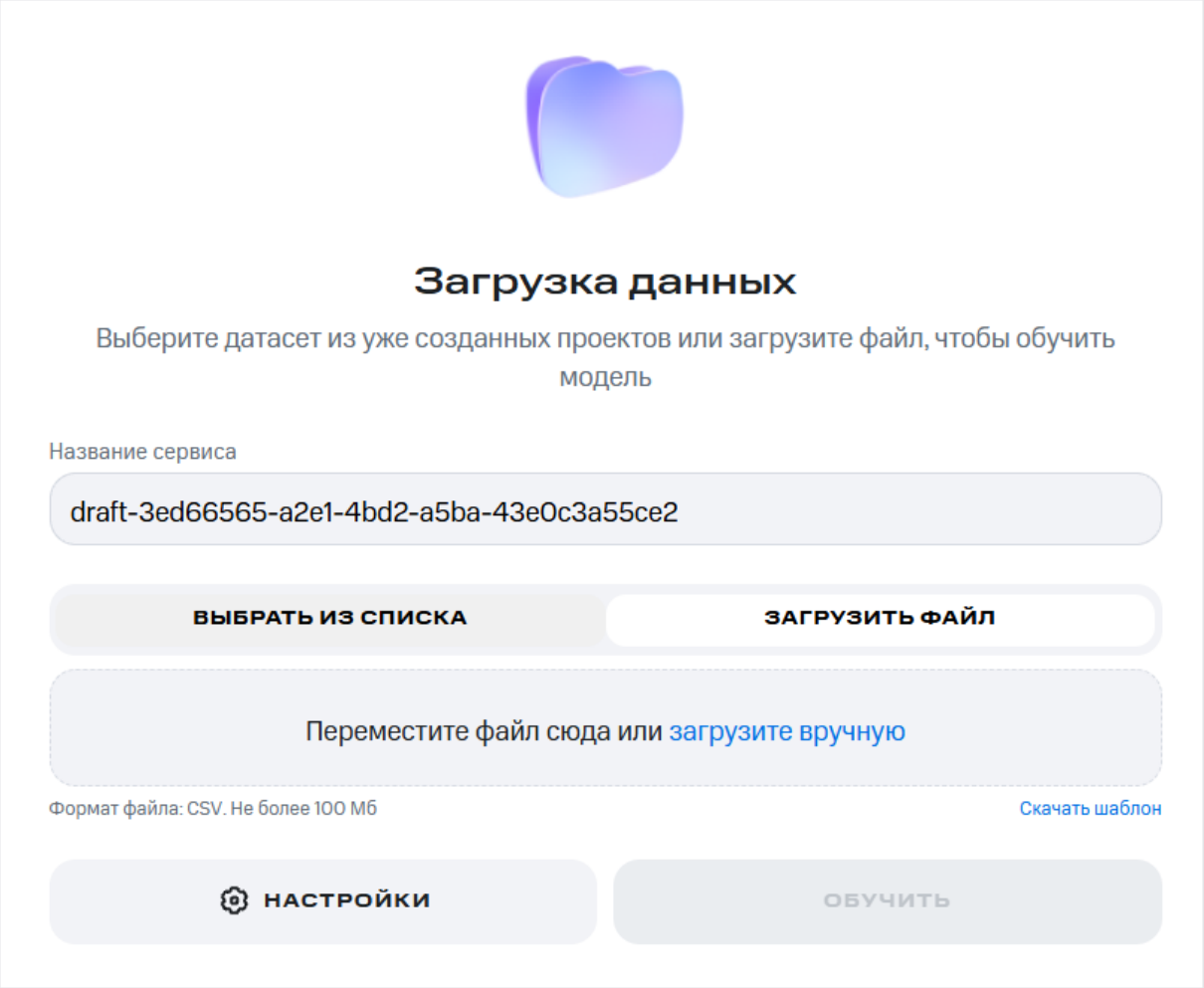
- Базы знаний — создание и управление БЗ
- RAG в сценариях — интеграция БЗ в диалоги

Привязка классификатора

Чтобы бот попадал в нужные сценарии на основе примеров, создайте сервис классификатора и привяжите его к сценарию. Для этого:

1. Создайте проект разметки данных и скачайте полученный датасет для обучения сервиса классификатора. Подробнее см. в разделе «Разметка данных».
2. В веб-клиенте перейдите на страницу **AI сервисы**.
3. Нажмите на кнопку **Создать сервис**.
4. В открывшемся окне выберите тип сервиса **Классификатор**.

5. В окне **Создание сервиса** измените название сервиса. По умолчанию в качестве названия сервиса используется 36-значный идентификатор. При необходимости его можно изменить позднее.



6. Загрузите сохраненный ранее датасет. Размер не должен превышать 100 МБ.
7. Нажмите на кнопку **Обучить**. В результате начинается процесс обучения модели. Он может занимать некоторое время. По окончании запуска готовый сервис переходит в статус **Запущен**.
8. Перейдите на страницу **Проекты** и откройте конструктор нужного бота.
9. На вкладке **Версии** откройте настройки текущей версии.
10. В выпадающем списке **Классификатор** выберите ранее обученный классификатор.

> **ВНИМАНИЕ**
>
> Для работы классификатора в сценарий препроцессинга необходимо добавить скрипт для вычисления результатов классификации. Проверьте, что в поле **Препроцессинг** указано значение **Required**. Подробнее см.в разделе «Интент».

11. Создайте версию из текущей и опубликуйте ее для дальнейшего размещения в канале.

Подробнее о сервисе классификатора см. раздел «AutoML-сервисы».

Удаление сценария

Не рекомендуется удалять сценарий целиком, так как на него могут вести ссылки из других сценариев. В этом случае появляется соответствующее предупреждение. Если нужно перезаписать сценарий, рекомендуется очистить только его содержимое. Для этого удалите блоки и связи, затем заново наполните сценарий логикой.

Если сценарий не нужен, и их из других сценариев нет на него ссылок, то вы можете удалить его. Для этого выделите его на вкладке **Сценарии**, вызовите контекстное меню по кнопке ••• и выберите пункт **Удалить**.

(пусто)

Развитие проекта

Бота можно обновлять и развивать, например, добавлять новые функции и исправлять ошибки. Все доработки выполняются в новой версии.

Для расширения возможностей бота в конструкторе доступны:

- обучение классификатора на основе размеченных диалогов. Обучите сервис классификатора, чтобы в дальнейшем бот выполнял запросы наиболее точно;
- использование сервиса NER для распознавания именованных сущностей в теле запроса;
- добавление управляющих элементов с произвольным кодом;
- использование управляющих элементов для HTTP-вызовов интеграций.

Подключение сервиса NER

Платформа поддерживает AutoML-сервисы, обслуживающие типы моделей Классификатор и **NER**.

NER – это ML-модель, способная распознавать именованные сущности в тексте запроса. Например, NER-сервис можно подключить в сценарий бота медицинского помощника для извлечения и структурирования информации. Из предложения «Появилась сыпь на руках и ногах» с помощью обученного сервиса определяются сущности «симптом» и «локация».

Чтобы использовать NER в сценариях:

1. Создайте сервис.
2. Протестируйте работу модели.
3. Подключите сервис к боту. Для этого используйте интеграционный блок HTTP-запрос. Укажите в нем URL созданного сервиса, а также заполните другие параметры.

Создание и обучение сервиса NER

1. Перейдите на страницу **AI-сервисы**. Нажмите на кнопку **Создать сервис**.
2. В открывшемся окне выберите тип сервиса **NER**. Нажмите на кнопку **Продолжить**.
3. Укажите имя сервиса и выберите датасет из созданных проектов или загрузите файл. Нажмите на кнопку **Обучить**, чтобы начать обучение.

Подробнее см. раздел «Создание AutoML-сервиса».

Тестирование сервиса NER

1. Откройте карточку сервиса.
2. Нажмите на кнопку **Тестировать**. В результате открывается панель **Тестирование**.

3. Введите запрос для тестирования и нажмите на кнопку  .

Подробнее см. раздел «Тестирование AutoML-сервиса».

Подключение сервиса к боту

1. Перейдите в конструктор сценариев бота.
2. Добавьте в сценарий блок **HTTP-запрос**. Заполните поля блока, например:

В параметрах укажите:

URL. Адрес сервиса NER. Скопируйте его из поля URL модели в карточке сервиса.

Метод. Выберите значение **POST**.

Headers. Укажите значение:

```
[{"key":"request-id","value":"{{system.system_message_id}}"},{"key":"Content-Type","value":"application/json"}]
```

Body. Укажите значение:

```
{
   "text": "string",
   "top_n": 1
}
```

Response mapping – маппинги.

Подробнее о заполнении полей см. в разделе «HTTP-запрос».

3. Наполните сценарий остальными блоками.

Использование программного кода

Если функциональности no-code-конструктора недостаточно для закрытия потребностей организации, то можно встроить в сценарий бота произвольный программный код. В конструкторе поддерживается код на языке Python. Для этого в рабочую область добавьте блок Скрипт.

(пусто)

Запуск проекта

1. Протестируйте созданный проект, чтобы проверить логику работы и убедиться в отсутствии ошибок в написании сценария. При необходимости исправьте ошибки.
2. Опубликуйте версию, если она еще не опубликована.
3. Подключите канал, по которому клиент может обратиться к боту и получить информацию.
4. Сделайте разметку истории диалогов по подключенному каналу, чтобы улучшить работу следующих запросов.

При необходимости ненужного бота можно удалить.

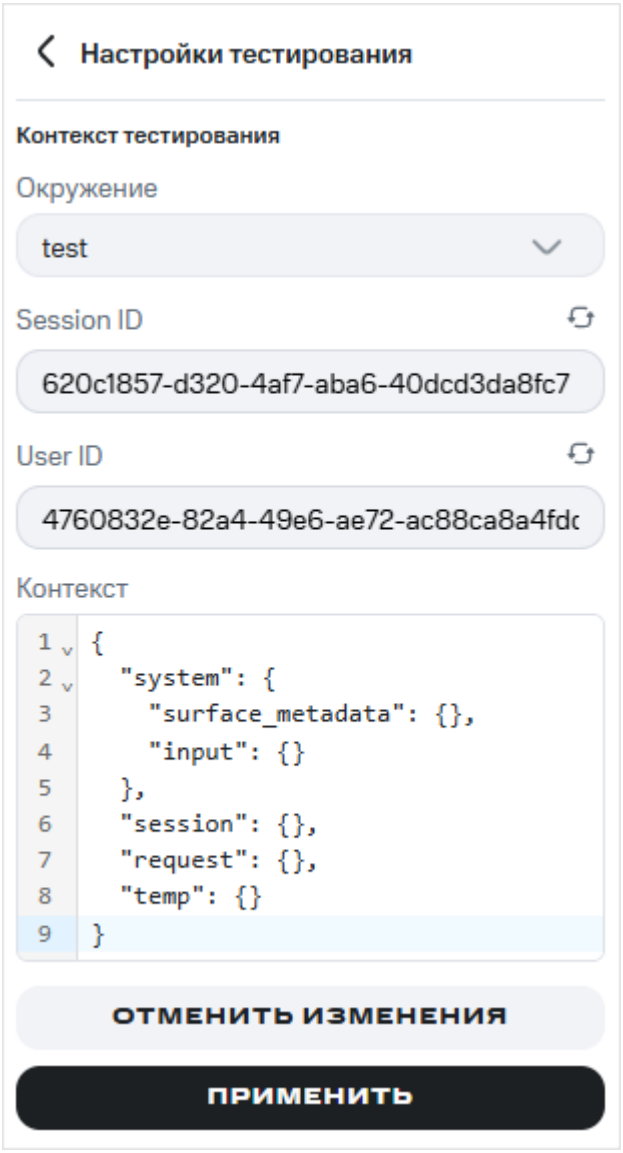
(пусто)

Тестирование проекта

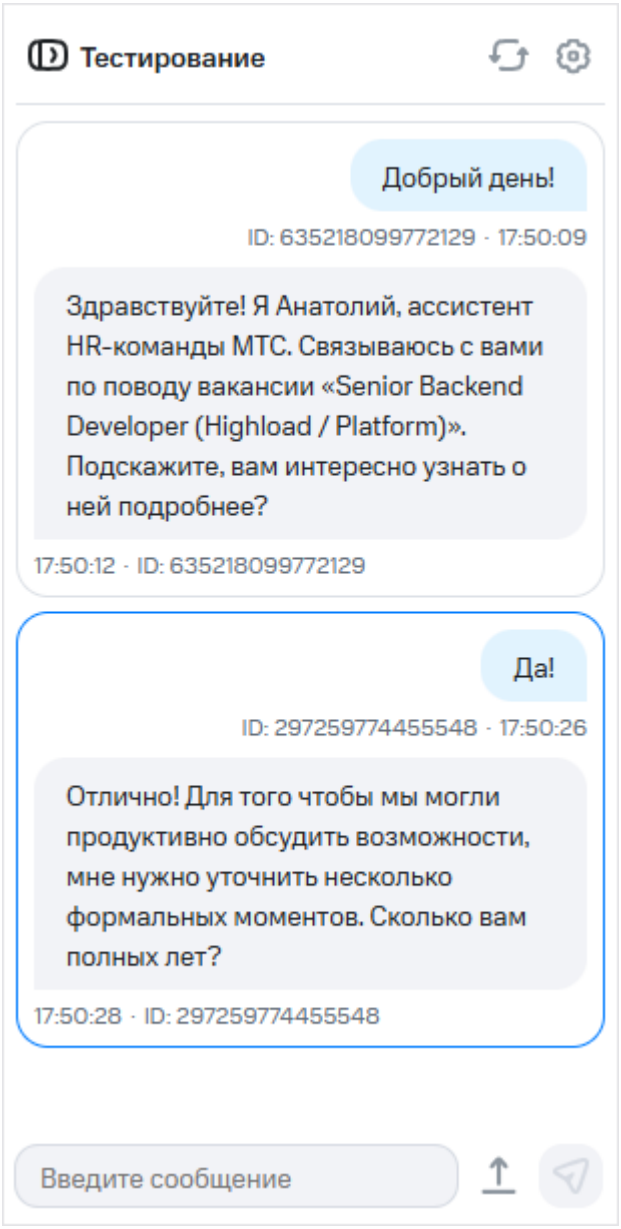
> **ВНИМАНИЕ**
>
> Перед началом тестирования убедитесь, что версия сохранена.

После разработки сценариев протестируйте бота. Для этого:

1. В конструкторе сценариев нажмите на кнопку **Тестирование**.
2. Настройте контекст тестирования. Для этого нажмите на кнопку  и скорректируйте нужные значения полей. Выберите окружение, для которого планируется протестировать сценарии. Затем нажмите на кнопку **Применить**.



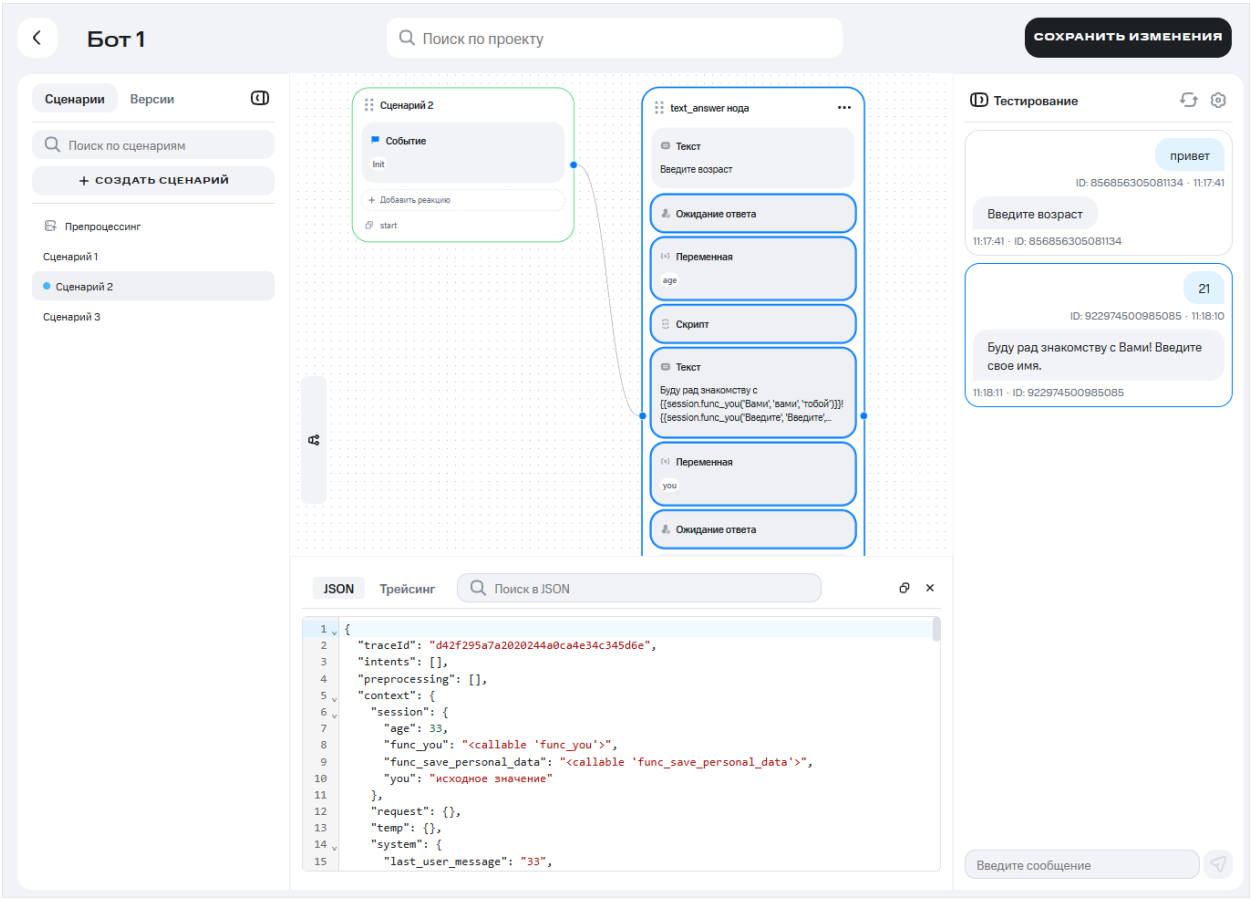
1. Введите тестовый запрос и проверьте корректность ответов.



ПОДСКАЗКА

Для эмуляции загрузки файлов в чат используйте кнопку . По умолчанию допустимый размер одного файла – 100 МБ.

В результате диалог с ботом отображается в виде чата с минимальной единицей – шаг диалога. Шаг представляет собой вопрос пользователя и ответ. Выделите шаг, чтобы посмотреть, какой сценарий был выбран и какие блоки отработали для запроса.



Для наглядности выбранные сценарии отмечаются в списке голубыми точками. Блоки, которые выполнились при обработке запроса, подсвечиваются синей рамкой в рабочей области конструктора.

Чтобы очистить диалог, нажмите на кнопку . Также диалог очищается при сворачивании тестового виджета.

Чтобы получить детальную информацию, проанализируйте данные на дебаг-панели на вкладках JSON и Трейсинг.

Техническая информация в формате JSON

Во вкладке отображаются «сырые» данные выполнения сценария в формате JSON, содержащие полную техническую информацию.

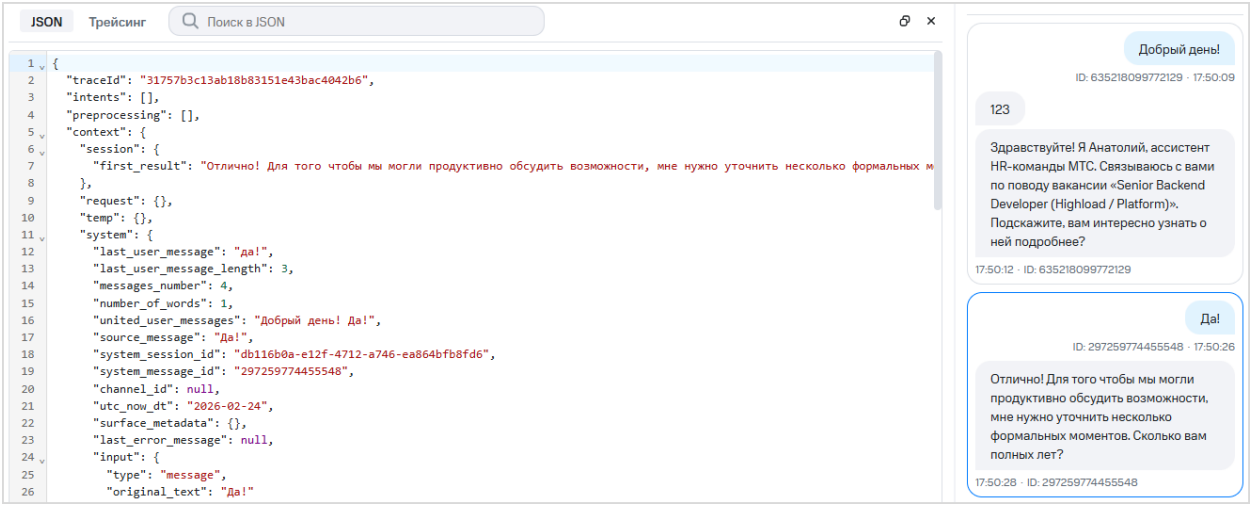
Основные возможности

- **Поиск по тексту** – ищите любые значения, имена полей, ошибки.
- **Копирование** – скопируйте весь JSON для анализа во внешних инструментах.
- **Техническая отладка** – полный доступ ко всем параметрам и контекстам.
- **Структура данных** – видна полная иерархия объектов и массивов.

Когда использовать

- Поиск специфических значений в контексте.

- Анализ технических ошибок и параметров.
- Изучение структуры данных сессии.
- Сохранение полного состояния для анализа.



Что отображается на вкладке JSON

Ключевые поля:

- **traceId** – уникальный идентификатор трассировки.
- **intents** – массив намерений, определенных классификатором бота (если используется).
- **preprocessing** – результаты препроцессинга сценария.
- **context** – контекст выполнения сессии, содержащий:
  - **session** – данные сессии пользователя
  - **request** – данные текущего запроса
  - **temp** – временные переменные
  - **system** – системные данные (метаданные контекста, счетчики)
  - **nlu** – результаты NLU анализа запроса (intents, rules, matches)
  - **events** – события и кандидаты
- **executions** – массив выполненных сценариев с узлами и блоками

Как работать с JSON

Поиск информации

1. Используйте поле поиска в верхней части JSON панели.
2. Введите текст для поиска (имя поля, значение, ошибка).
3. Система подсветит все совпадения в JSON и выведет их общее количество.
4. Используйте кнопки ‹ › для навигации между результатами.

Копирование данных

1. Нажмите на иконку в верхней правой части панели.
2. Весь JSON будет скопирован в буфер обмена.

3. Можно вставить в любой текстовый редактор для анализа.

Навигация по JSON

1. Чтобы свернуть/развернуть секции JSON (объекты и массивы), используйте стрелки на панели нумерацией строк JSON.
2. Различные типы данных подсвечены разными цветами.
3. JSON автоматически отформатирован для удобного чтения.

Анализ выполнения блоков

1. Откройте JSON вкладку.
2. Найдите секцию **execution.**
3. Изучите структуру выполнения:

Пример структуры **execution**:


```
{
  "executions": [
    {
      "scenarioId": 2491,
      "scenarioName": "scenario2",
      "activation": {
        "type": "",
        "score": 0.0,
        "value": ""
      },
      "nodes": [
        {
          "nodeId": "9dfda162-fc06-43d8-8292-2ea8d35d69e9",
          "nodeName": "node 1",
          "blocks": [
            {
              "blockId": "b99af4cc-2d31-4ec4-a11b-5fca63899f93",
              "result": {},
              "timestamp": "2025-01-01T00:00:00Z"
            }
          ]
        }
      ]
    }
  ]
}
```


Ключевые поля:

- **scenarioId** и **scenarioName** – ID и название сценария.
- **nodeId** и **nodeName** – ID и название узла.
- **blockId** – ID выполненного блока.
- **result** – результат выполнения (может быть пустым объектом).
- **timestamp** – время выполнения.

Поиск ошибок

1. Откройте вкладку JSON.
2. Выполните поиск по слову «error».
3. Проверьте поля в execution-блоках:
   - **error** – информация об ошибке
   - **errorMessage** – текст ошибки
   - **isError: true** – флаг наличия ошибки

Пример с ошибкой:

```
{
  "blockId": "script_block_001",
  "result": null,
  "error": {
    "type": "ScriptError",
    "message": "Variable 'undefined_var' is not defined",
    "line": 15
  },
  "timestamp": "2026-02-26T11:30:00Z"
}
```

Трейсинг

Трейсинг – это детальное отслеживание этапов выполнения сценария, с помощью которого получают:

- последовательность исполнения блоков сценария;
- таймлайн – время обработки каждого шага;
- места возникновения ошибок;
- детали выполнения: входные/выходные данные, параметры вызовов LLM, инструментов, HTTP-запросов.

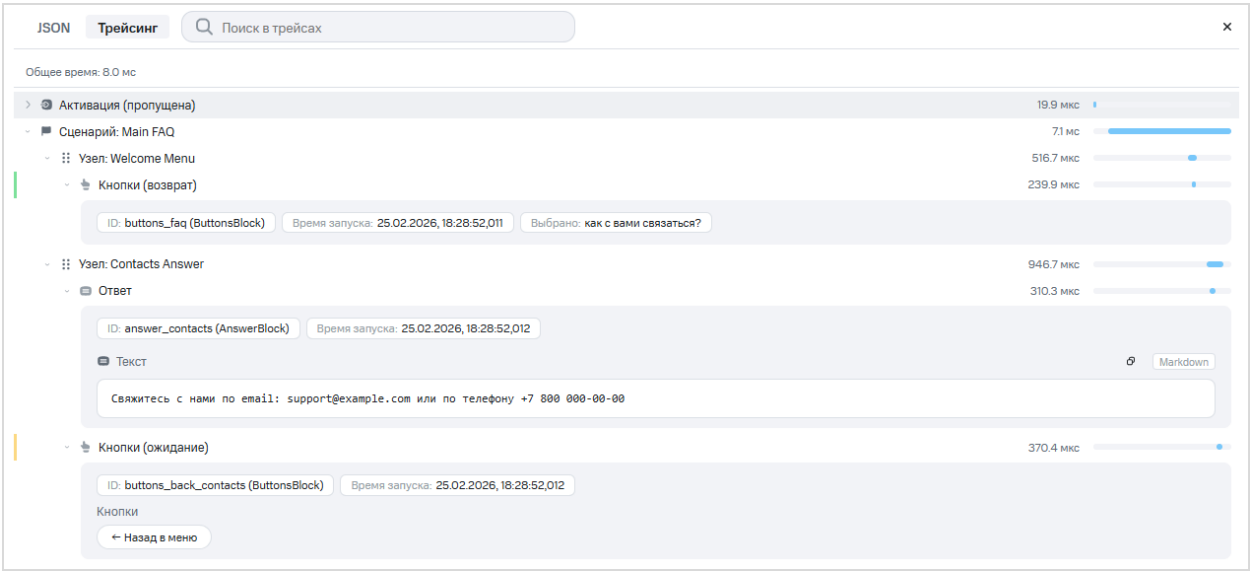
Основные возможности

- **Визуализация** – иерархическое дерево выполнения с иконками.
- **Временные метрики** – таймлайны показывают время выполнения каждого блока сценария.
- **Визуальные индикаторы** – ошибки, прерывания, продолжения сценария подсвечиваются цветом .
- **Навигация к блокам** – двойной клик переходит к блоку в рабочей области конструктора сценариев.
- **Читаемость входных и выходных данных** – форматирование Markdown.

Когда использовать

- Быстрый поиск ошибок.

- Оптимизация производительности по данным таймлайнов.
- Понимание логики выполнения сценария.
- Анализ входных/выходных данных, HTTP-запросов, LLM-взаимодействий и инструментов.
- Интерактивная навигация по сценарию.



Что отображаетсяв трейсинге

Дерево запроса

- Иерархическое представление элементов: **Сценарий** → **Узлы** → **Блоки** → **Вызовы**;
- Каждый уровень предоставляет свою информацию о выполнении.

Визуальные индикаторы событий

- **Ошибка** – красный фон строки и таймлайна;
- **Interrupt** (ожидание пользователя) – жёлтая полоса слева от строки;
- **Resume** (продолжение) – зелёная полоса слева от строки.

Как работать с трейсингом

Поиск информации

1. Используйте поле **Поиск в трейсах.**
2. Введите текст для поиска (например имя блока, значение или текст ошибки).
3. Система подсветит все совпадения в трейсе и выведет их общее количество.
4. Используйте кнопки ‹ › для навигации между результатами.

Просмотр детальной информации блока

1. Клик на узле разворачивает или сворачивает состав блоков.
2. Кликните на любой блок – раскроется панель с деталями.
3. Двойной клик на блоке – переход к блоку на рабочем поле.

Детали блока содержат

- Ключевые параметры.
- Параметры генерации LLM – temperature, max_tokens, top_p и др.
- Входные и выходные данные в зависимости от типа блока: промпты, тексты ответов, входы/выходы инструментов.
- Переключатель Raw/Markdown для выбора формата отображения ответов в блоке.

Практическое использование трейсинга

Поиск ошибокв сценарии

1. Оправьте тестовое сообщение боту.
2. Откройте вкладку **Трейсинг** дебаг-панели.
3. Ищите элементы с красным фоном.
4. Нажмите на ошибочный блок, чтобы увидеть детали.
5. Проанализируйте, что пошло не так:
    - неверные входные данные;
    - ошибка в LLM-ответе;
    - проблема с вызовом инструмента;
    - ошибка HTTP-запроса.

Оптимизация производительности медленного сценария

1. Обратите внимание на таймлайны блоков – они показывают время исполнения.
2. Ищите блоки с самыми длинными таймлайнами.
3. Разверните «медленные» блоки, чтобы посмотреть детали.
4. Проверьте:
    - слишком длинные промпты у LLM;
    - слишком объемные ответы модели;
    - медленные ответы внешних сервисов.

Отладка сложных сценариев

1. Следуйте по трейсу сверху вниз.
2. Используйте поиск. чтобы найти конкретные блоки по имени.
3. Двойной клик на блок поможет быстро найти его в конструкторе.
4. Сравните реальный маршрут выполнения сценария с ожидаемым.

Что делать, если трейсы не появляются

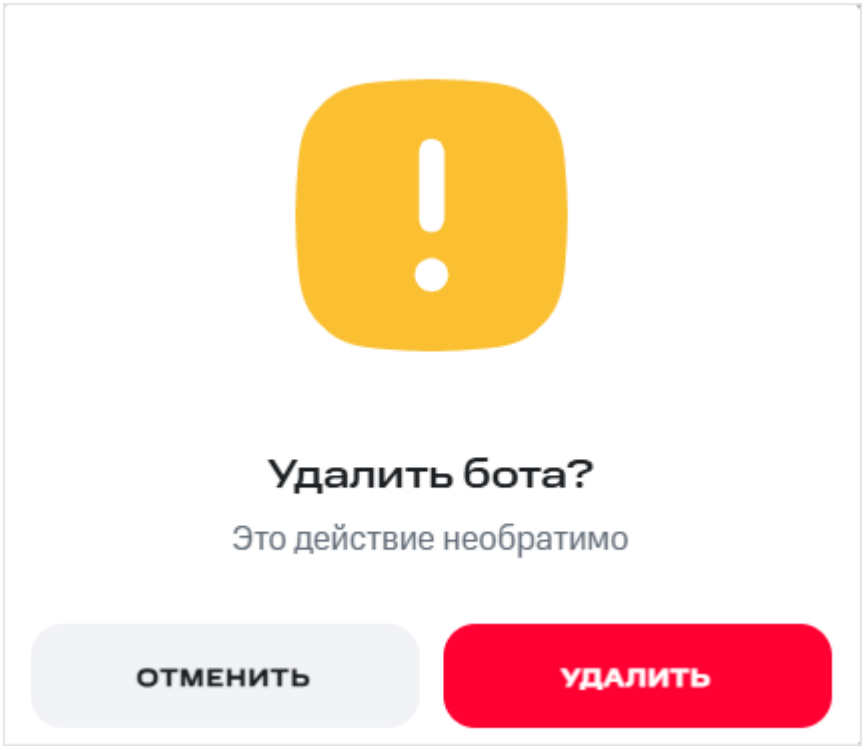
1. Подождите 5-10 секунд – данные загружаются с небольшой задержкой.
2. Проверьте, что вы отправили сообщение через виджет тестирования.

Удаление проекта

Чтобы удалить неиспользуемый проект:

1. Перейдите на страницу **Проекты**.

2. По кнопке ••• вызовите контекстное меню нужного бота и выберите пункт **Удалить**.
3. В появившемся диалоговом окне подтвердите удаление.



4. В списке проектов проверьте, что удаленного проекта в нем нет.

(пусто)

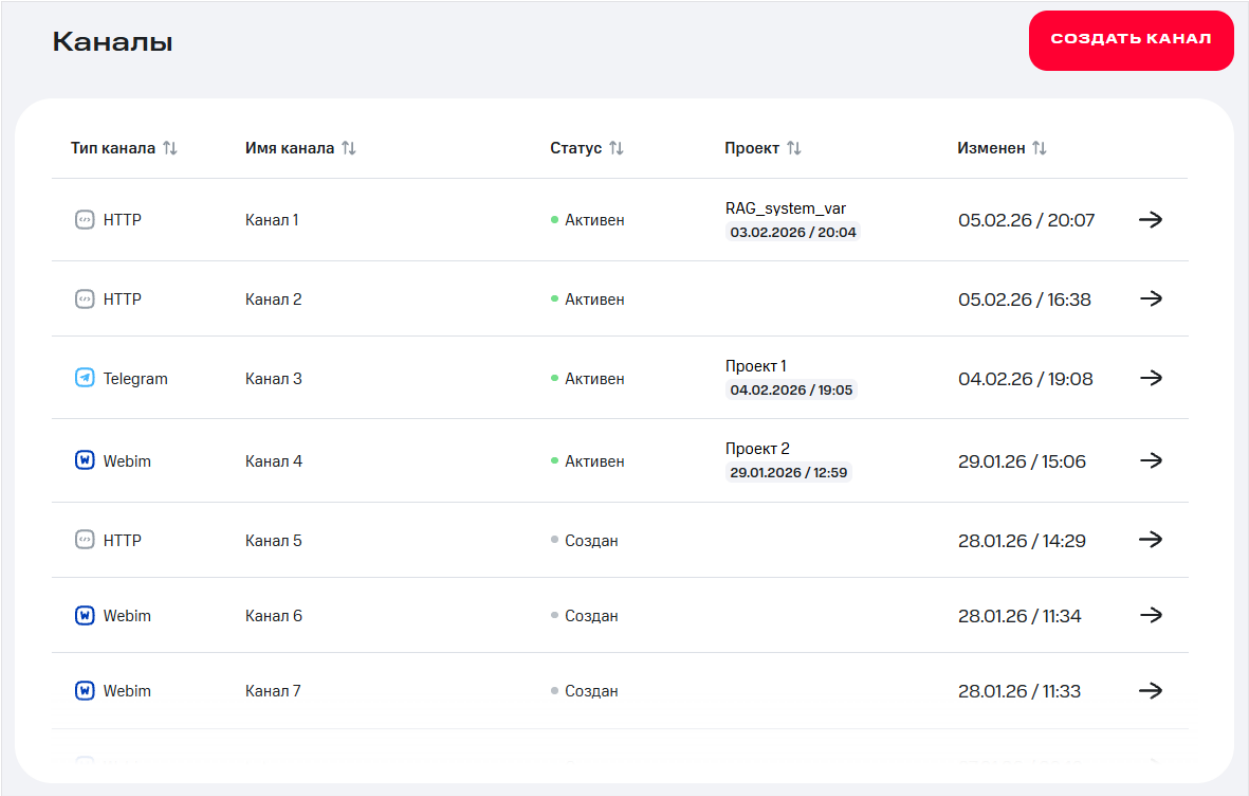
Публикация проекта в канале

Чтобы клиенты могли общаться с ботом, его нужно разместить в канале. Канал — это сущность, которая соединяет бота с пользователем, будь то мессенджер или чат на сайте. В одном канале работает только один бот, но бот может быть доступен в нескольких каналах одновременно.

Платформа поддерживает каналы для следующих типов поверхностей:

- Webim – это омниканальная система для общения с клиентами. Вы можете упростить и автоматизировать коммуникационные процессы в Webim, если интегрируете с ней сценарий вашего бота.
- Telegram – канал обеспечивает прием и отправку текстовых и звуковых сообщений, файлов, изображений, документов в мессенджере Telegram. Для интеграции используется Telegram Bot API;
- HTTP – поддерживает прием и отправку данных по HTTP -протоколу, например чаты на сайтах, виджеты, умные устройства;
- MAX – канал для интеграции бота с мессенджером MAX.

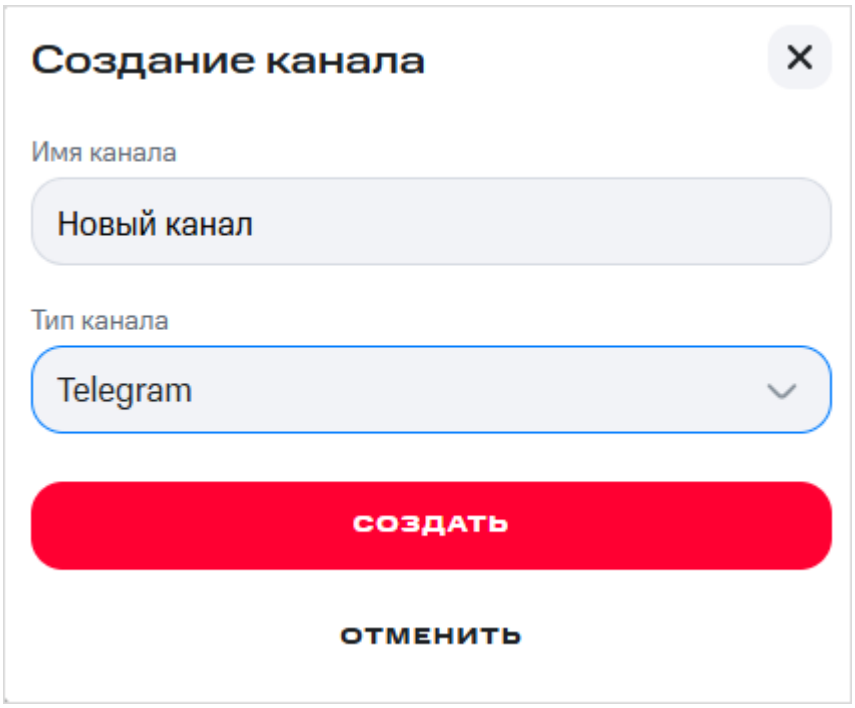
Список каналов и основная информация о них, включая статус, отображается в разделе **Каналы**. Чтобы изменить настройки какого-либо канала из списка, перейдите в его карточку по кнопке →.



Создание канала

Чтобы создать новый канал для публикации бота:

1. Перейдите в раздел **Каналы**.
2. Нажмите на кнопку Создать канал.

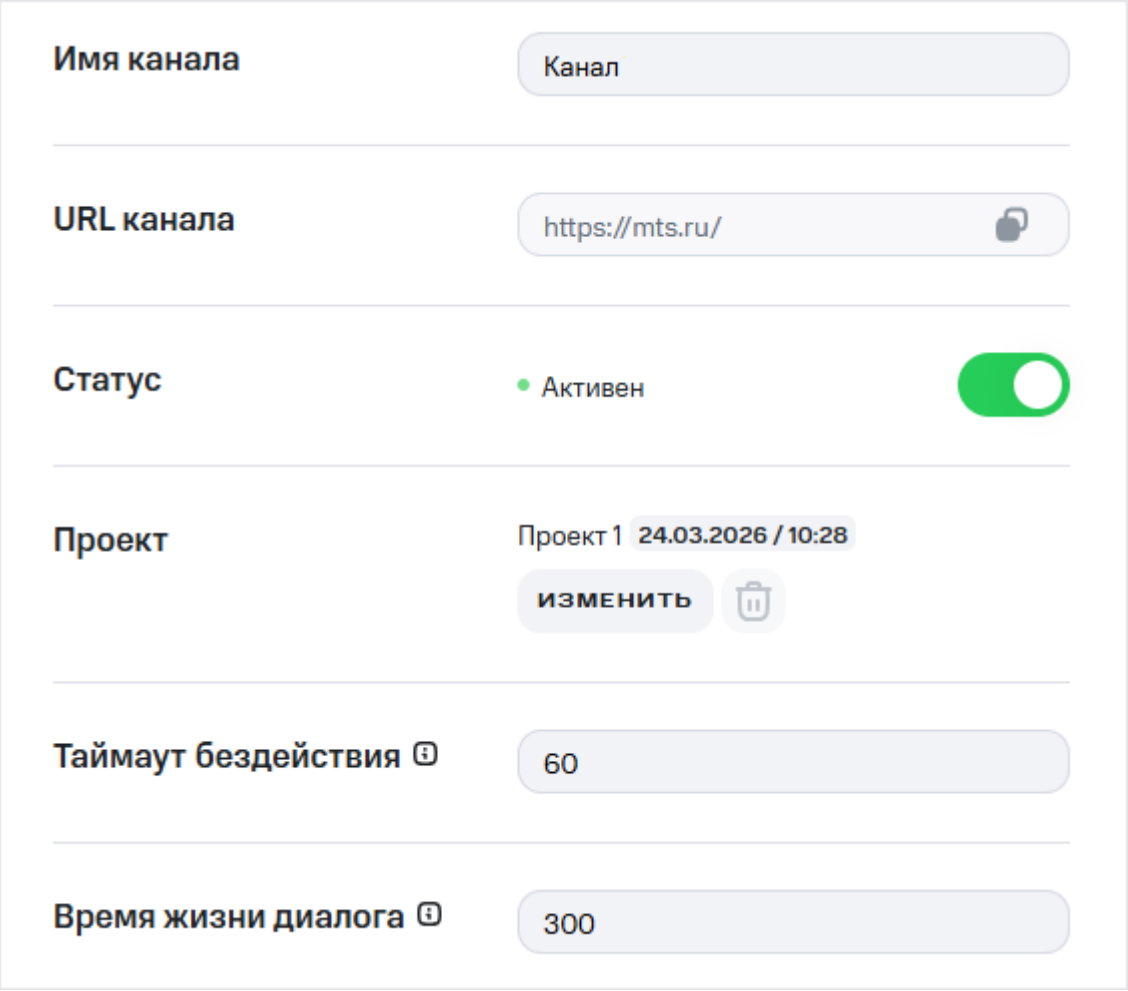


3. В окне **Создание канала** задайте имя и тип, затем нажмите на кнопку **Создать** – откроется карточка канала выбранного типа. Задайте настройки в зависимости от типа канала.

> **ВНИМАНИЕ**
>
> Тип созданного канала нельзя изменить.

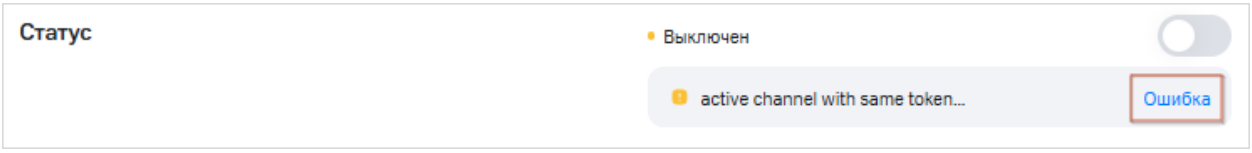
Настройка канала

Базовые настройки идентичны для всех типов каналов.



Имя канала. Уникальное название канала в системе. **URL канала**. Генерируемый системой адрес, по которому бот будет доступен после активации канала. Поле не редактируется. **Статус**. Кнопка включения/выключения канала. Статус канала меняется в следующем контексте:

- **Создан** – стартовый статус канала при создании.
- **Активируется** – канал в процессе запуска.
- **Активен** – канал работает.
- **Выключается** – канал в процессе выключения.
- **Выключен** – канал выключен. Этот статус присваивается в том числе и при выключении канала из-за ошибки в работе или при активации канала (например, когда истек срок жизни токена или токен уже используется в другом канале). В этом случае отображается информация о причинах выключения. Полный текст ошибки доступен по одноименной ссылке в карточке канала.



Проект. По кнопке **Выбрать** открывается окно для выбора бота и его версии для размещения в канале.

> **ВНИМАНИЕ**
>
> Для выбора доступны только версии, которые опубликованы. Подробнее см. раздел «Версии».

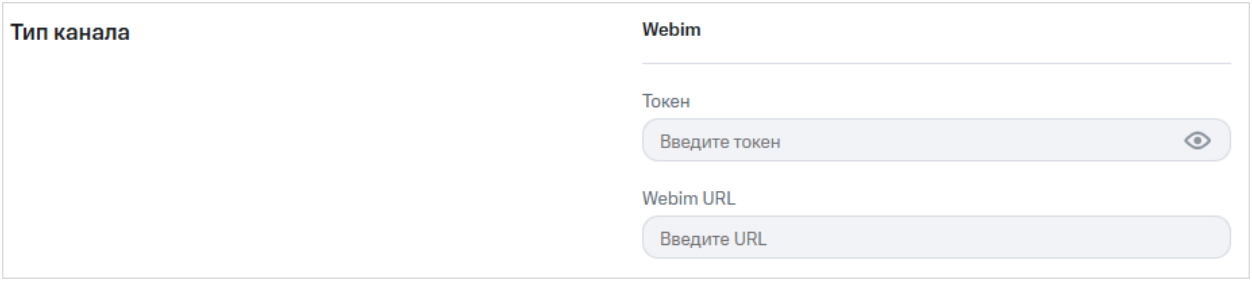
Таймаут бездействия. Время, в течение которого пользователь неактивен. По истечении этого времени наступает событие **Inactivity Timeout**, по которому активируется соответствующий сценарий. Указанный таймаут должен быть меньше времени жизни диалога. Если таймаут указан, а соответствующего сценария нет, то значение параметра игнорируется. Подробнее см. в разделе «Событие».

Время жизни диалога. Время существования сессии диалога с пользователем после последнего пользовательского запроса. По умолчанию – **300** секунд.

Чтобы создать работоспособный канал, установите все перечисленные базовые настройки. Затем в зависимости от типа канала заполните перечисленные ниже дополнительные поля в карточке.

Webim

Для канала Webim необходимо заполнить поля **Токен** и **Webim URL**.



Telegram

Для Telegram укажите:

Токен для регистрации канала в Telegram – его необходимо получить с помощью telegram-бота BotFather (ссылка указана в карточке канала).

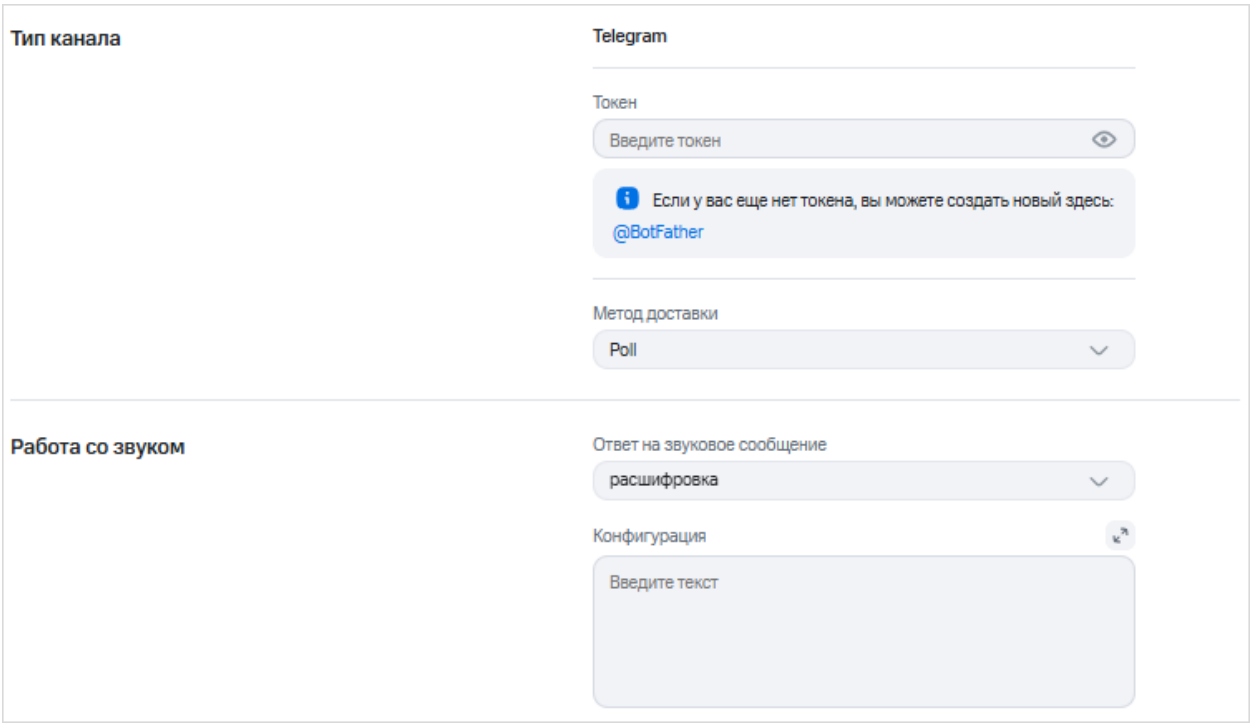
Метод доставки:

- **POLL** – бот опрашивает telegram-сервер с определенным интервалом. Если есть новые сообщения, они возвращаются в ответе. Подходит для тестирования и небольших проектов с низким трафиком, где допустимы минимальные задержки в отклике.
- **PUSH** – telegram-сервер сам отправляет, «пушит» новые сообщения в реальном времени на URL канала. Подходит для ботов с высоким трафиком, где важны мгновенные обновления и масштабируемость.

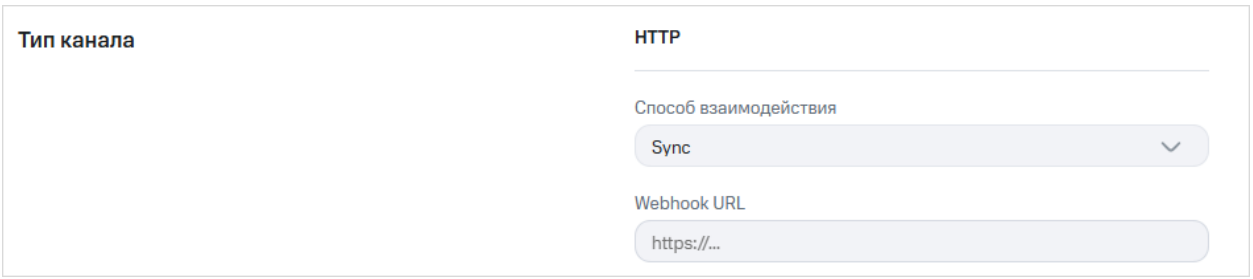
Работа со звуком. Настройки передачи аудио в сообщениях:

- **Ответ на звуковое сообщение**:
  - расшифровка;
  - звук;
  - звук+расшифровка;

Конфигурация STT (Speech-to-Text) и TTS (Text-to-Speech) задается в одноименном поле при помощи JSON. Если оставить поле пустым будут использованы настройки, заданные по умолчанию при установке платформы. Подробнее см. раздел «Конфигурация аудио-сообщений в Telegram».



HTTP



Для HTTP-канала определите **Способ взаимодействия**:

- **Sync (Synchronous)** – классический »запрос-ответ» способ, где клиент – платформа отправляет запрос к поверхности (http-сервер) и блокируется (ждёт), пока не получит полный ответ. Подходит для тестовых или простых, низкотрафиковых ботов.
- **SSE (Server-Sent Events)** – способ для односторонних push-уведомлений от поверхности (http-сервера) к платформе в реальном времени. Подходит для систем, требующих оперативного, но не всегда полного ответа, который можно дослать в процессе взаимодействия. Например, в чатах с эффектом «печатания», когда бот отвечает, набирая текст по словам (например, Chat-GPT) или умных колонках, которые озвучивают ответ поэтапно.

- **Async (Asynchronous)** – асинхронный режим с отправкой ответа на внешний Webhook URL. Клиент отправляет запрос и мгновенно получает подтверждение (HTTP 202). Ответ бота приходит позже на указанный Webhook URL. Используйте **Async**, если:
  - сценарий бота выполняется долго (RAG, цепочки LLM-вызовов, обращения к внешним API).
  - клиент не может держать соединение открытым.
  - вы интегрируете бота в стороннюю систему (CRM, мобильное приложение, мессенджер).

В теле ответа на запрос клиента адаптер отправляет:

ПОЛЕ	ОПИСАНИЕ
status	accepted – запрос принят
sessionId	Идентификатор сессии. Передайте его в следующем запросе, чтобы продолжить диалог
externalRequestId	Идентификатор из исходного запроса. Используйте для сопоставления запроса и ответа

Webhook URL

URL, на который адаптер отправляет ответы бота методом POST.

Для режима **ASYNC** – это обязательное поле. Все ответы бота доставляются на этот адрес.

Для режимов **SYNC** и **SSE** – Webhook URL вводить необязательно. Для этих режимов он работает как резервный канал доставки: адаптер отправит ответ на Webhook URL, когда активное соединение с клиентом отсутствует.

> **ВНИМАНИЕ**
>
> Без Webhook URL в режимах SYNC и SSE ответы бота, отправленные при отсутствии соединения, будут потеряны. Например, если сработал таймаут неактивности, а пользователь уже закрыл страницу.

РЕЖИМ	WEBHOOK URL	ЧТО ПРОИСХОДИТ БЕЗ НЕГО
ASYNC	Обязателен	Канал не активируется
SSE	Опционален	Ответы без активного соединения теряются
SYNC	Опционален	Ответы без активного соединения теряются

Пример ASYNC-диалога

1. Клиент отправляет сообщение в канал.
2. Адаптер возвращает HTTP 202 с sessionId.
3. Бот обрабатывает запрос и отправляет ответ на Webhook URL.
4. Клиент отправляет следующее сообщение с тем же sessionId – диалог продолжается.

Пример Webhook как резервный канал для SYNC/SSE

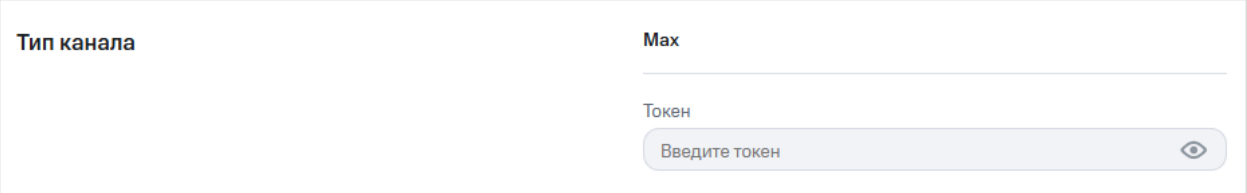
1. Пользователь написал боту и получил ответ.
2. Пользователь закрыл страницу – соединение разорвано.

3. Сработал таймаут неактивности — бот отправляет сообщение.
4. Адаптер не находит активное соединение и доставляет ответ на Webhook URL.

> **ПОДСКАЗКА**
>
> Если вы создаете собственный HTTP-клиент, API HTTP-адаптера платформы можно найти в руководстве администратора, раздел «Методы HTTP-adapter».

Max

Заполните поле Токен:



Управление настройками канала

В процессе работы канала может понадобиться внести изменения в его настройки.

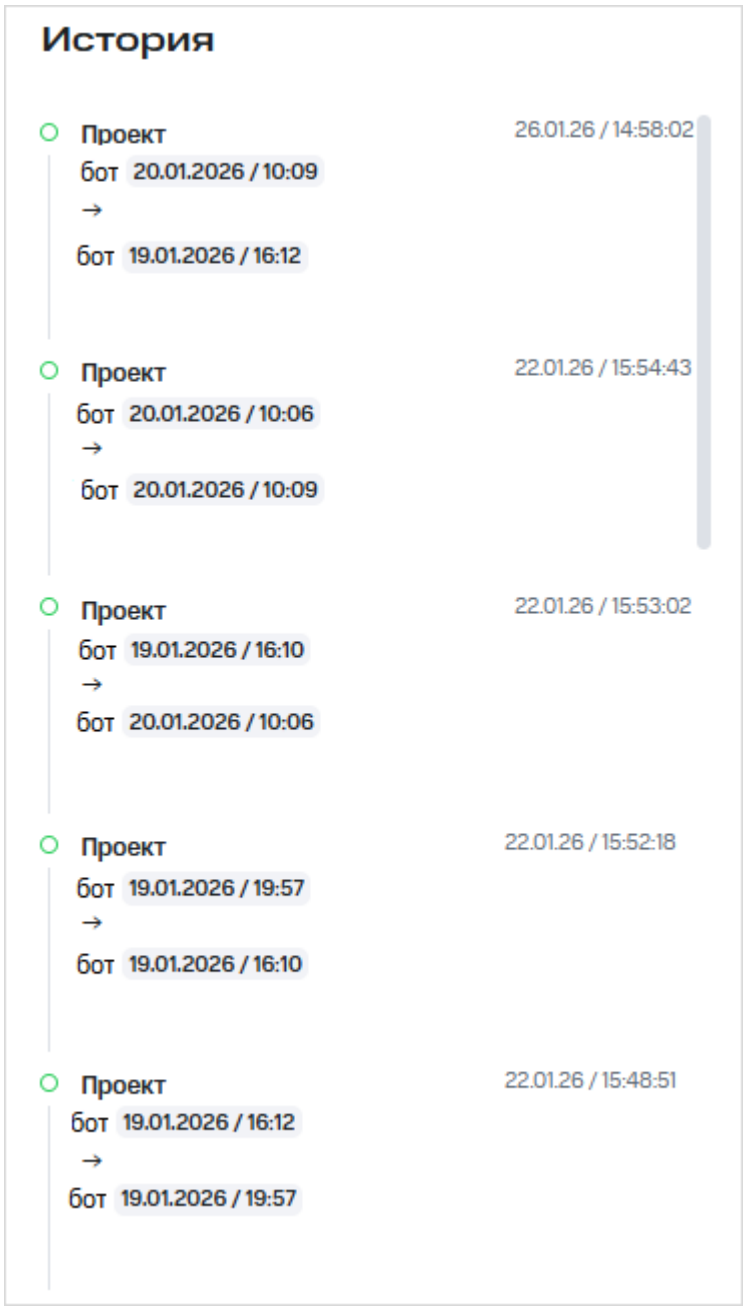
Часть настроек всегда доступна для редактирования (если канал в статусе **Активен**, его не нужно выключать):

- **Имя канала**.
- **Проект** — выбор нового бота, сценария.
- **Таймаут бездействия**.
- **Время жизни**.

Чтобы изменить токен, метод доставки или способ взаимодействия, а также, чтобы удалить бота из канала, канал необходимо перевести в статус **Выключен**.

История канала

Все изменения канала отображаются в его карточке в разделе **История**:



Удаление канала

При необходимости канал можно удалить. Для этого в карточке канала нажмите на кнопку **Удалить**.

Конфигурация аудио-сообщений в Telegram

Настройки обработки аудио-сообщений для канала Telegram задаются в поле **Конфигурация** при помощи JSON, содержащего конфигурацию механизмов STT (Speech-to-Text, распознавание речи) и TTS (Text-to-Speech, синтез речи). В качестве обработчика звука платформа использует продукт Audiogram.

> **ВНИМАНИЕ**
>
> В разделе приведен пример JSON и описание базовых параметров. Более подробную информацию о настройке STT и TTS вы можете найти в официальной документации Audiogram https://mts.ai/ru/product/audiogram/audiogram-doc/.

Пример JSON с базовой конфигурацией:

```
{
  "STT": {
    "encoding": 1,
    "sample_rate_hertz": 8000,
    "language_code": "ru",
    "audio_channel_count": 1,
    "model": "e2e-v3",
    "va_config": {
      "usage": 1
    },
    "punctuation_config": {
      "enable": true
    },
    "denormalization_config": {
      "enable": true
    }
  },
  "TTS": {
    "language_code": "ru",
    "encoding": 1,
    "sample_rate_hertz": 8000,
    "voice_name": "gavrilov",
    "synthesize_options": {
     "model_type": "high_quality",
      "voice_style": 0
    }
  }
}
```

Параметры STT (Speech-to-Text)

ПАРАМЕТР	ОПИСАНИЕ	ВОЗМОЖНЫЕ ЗНАЧЕНИЯ
encoding	Формат кодирования аудио	1 – LINEAR_PCM (PCM). WAV linear PCM аудиофайл с заголовком, содержащий целые знаковые 16-битные сэмплы в линейном распределении (PCM 16bit) и заданной частотой дискретизации в соответствии с полем sample_rate_hertz 3 – MULAW. WAV PCM аудиофайл с заголовком, содержащий 8-битные сэмплы в формате mu-law и заданной частотой дискретизации в соответствии с полем sample_rate_hertz 20 – ALAW. AV PCM аудиофайл с заголовком, содержащий 8-битные сэмплы в формате a-law и заданной частотой дискретизации в соответствии с полем sample_rate_hertz.
sample_rate_hertz	Частота дискретизации модели (в герцах). Если указана частота дискретизации отличная от значений, поддерживаемых моделью, то: - в случае распознавания речи (ASR) произойдет перекодирование частоты дискретизации на значение, поддерживаемое моделью (16000 Гц). - в случае синтеза речи (TTS) будет использована модель с ближайшей частотой дискретизации в большую сторону.	Число, обычно 8000, 16000, 44100, 48000 Гц
language_code	Код языка аудио для распознавания. По умолчанию ru	Строка, например "ru", "en", "kk"
audio_channel_count	Количество аудиоканалов	Целое число, обычно 1 (моно) или 2 (стерео)
model	Модель распознавания речи	e2e-v3 (sample_rate = 16000 Гц)

ПАРАМЕТР	ОПИСАНИЕ	ВОЗМОЖНЫЕ ЗНАЧЕНИЯ
va_config.usage	Выбор алгоритма обнаружения голоса	0 – стандартное обнаружение голоса 1 – DO_NOT_PERFORM_VOICE_ACTIVITY (без обнаружения голоса) 2 – USE_DEP (использовать DEP алгоритм) 3 – USE_ENHANCED_VAD (улучшенное обнаружение голоса) 4 – USE_TARGET_SPEECH_VAD (целевое обнаружение речи)
punctuation_config.enable	Включение автоматической пунктуации	true – включить false – отключить
denormalization_config.enable	Включение денормализации текста	true – включить false – отключить

Параметры TTS (Text-to-Speech)

ПАРАМЕТР	ОПИСАНИЕ	ВОЗМОЖНЫЕ ЗНАЧЕНИЯ
language_code	Код языка для синтеза речи	Строка, например "ru", "en"
encoding	Язык, используемый в аудиофайле	1 – LINEAR_PCM (PCM) 3 – MULAW 20 – ALAW
sample_rate_hertz	Частота дискретизации синтезированного аудио в герцах	Число, обычно 8000, 16000, 22050, 24000, 44100, 48000 Гц
voice_name	Имя голоса для синтеза речи	Строка с названием голоса, зависит от доступных голосов на сервере
synthesize_options.model_type	Тип модели синтеза	Строка, возможные значения зависят от сервера, например "high_quality", "standard"
synthesize_options.voice_style	Стиль голоса	0 – нейтральный 1 – радостный 2 – злой 3 – грустный 4 – удивленный 5 – разговорный

ПОДСКАЗКА

Рекомендации по настройкам:

- Частота 8000 Гц оптимальна для голосовых сообщений в Telegram (баланс между качеством и размером файла).
- Модель "e2e-v3" – современная модель распознавания речи с использованием сквозных (end-to-end) нейронных сетей.
- Голос "gavrilov" – популярный русскоязычный мужской голос.
- При использовании голосовых сообщений рекомендуется установить encoding = 1 (LINEAR_PCM) для наилучшей совместимости.
- При необходимости улучшения качества распознавания можно включить дополнительные опции в va_config.

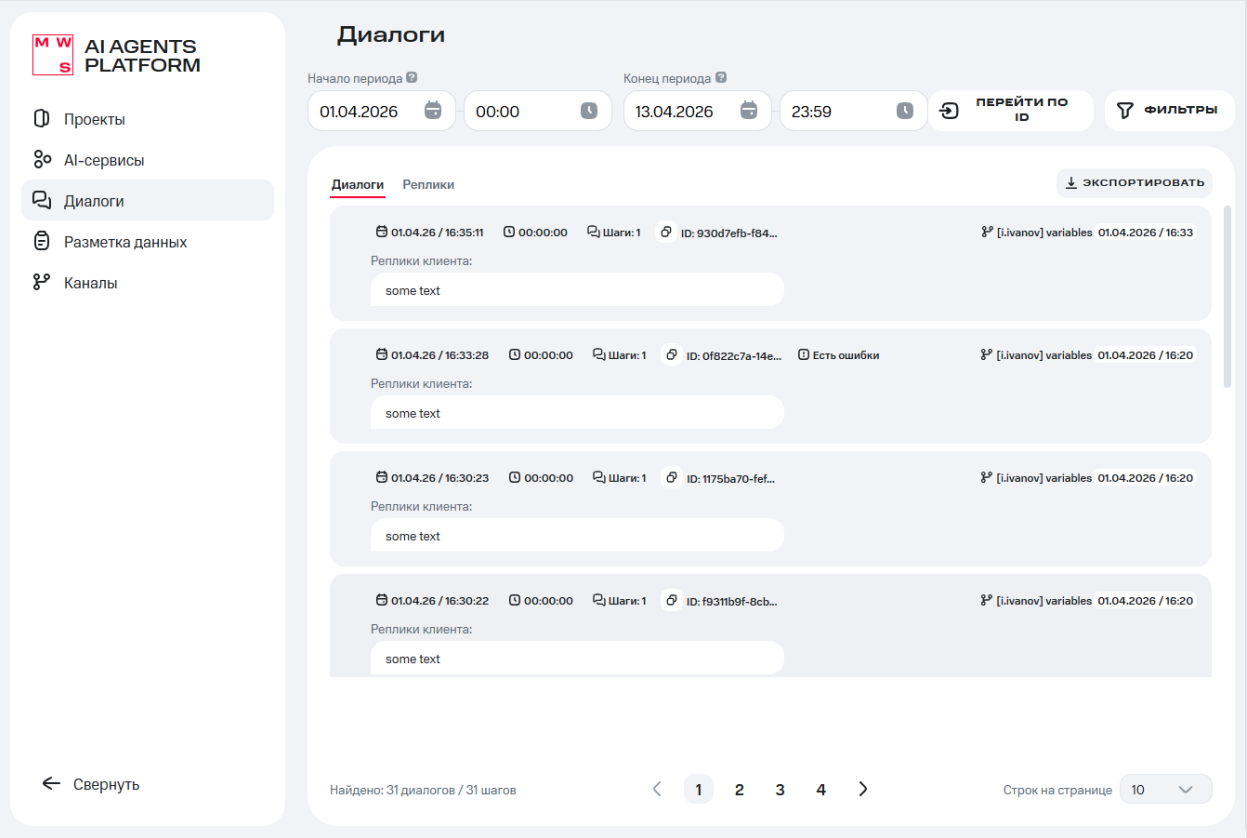
Диалоги

Чтобы в дальнейшем бот отвечал на вопросы более точно, рекомендуется регулярно размечать диалоги, которые уже велись с пользователями.

Для этого разметьте диалоги, затем добавьте файл с результатами к существующему проекту разметки.

Разметка диалога

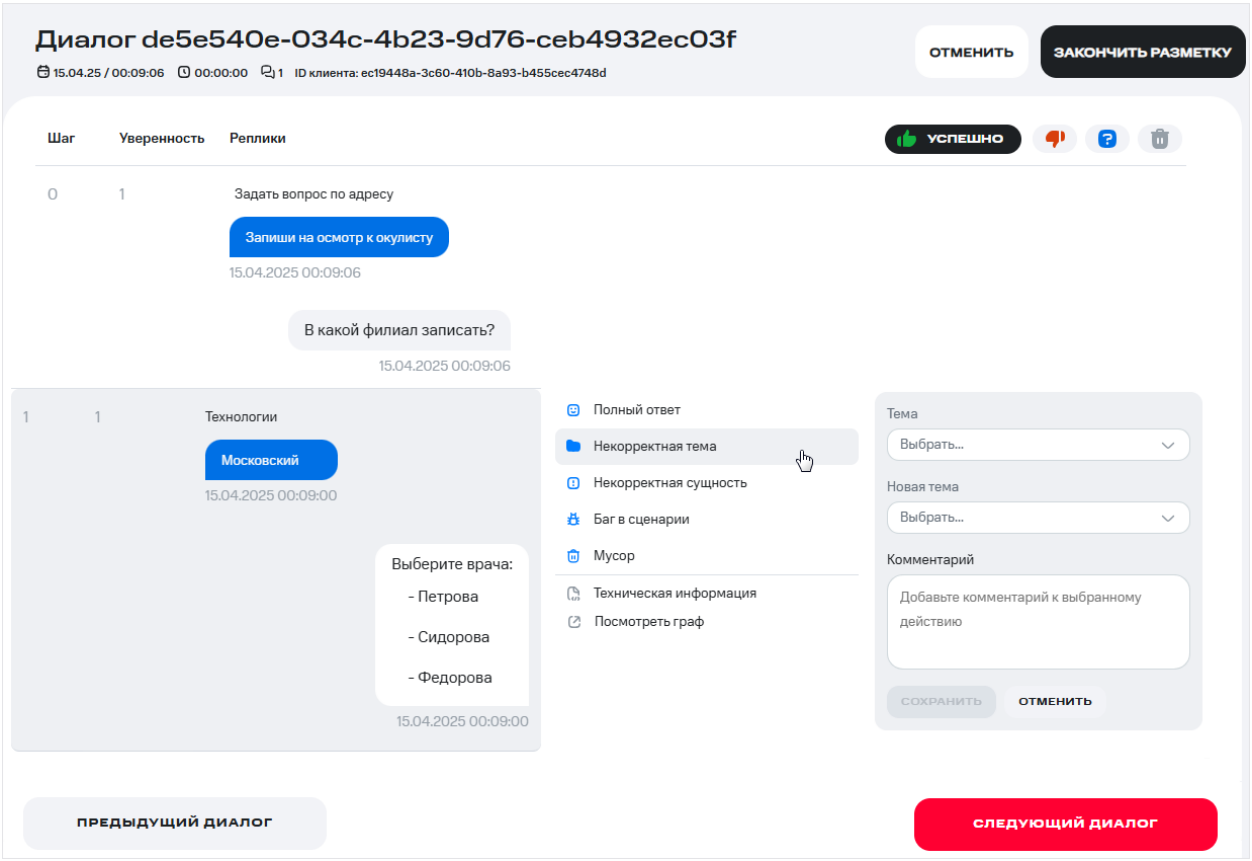
В веб-клиенте откройте страницу **Диалоги**.



1. Найдите нужный диалог. Для удобного поиска вы можете указать временной промежуток, в который состоялся диалог, а также использовать кнопки **Перейти по ID** и **Фильтры**.
2. Откройте диалог. Для этого нажмите на строку с ним.

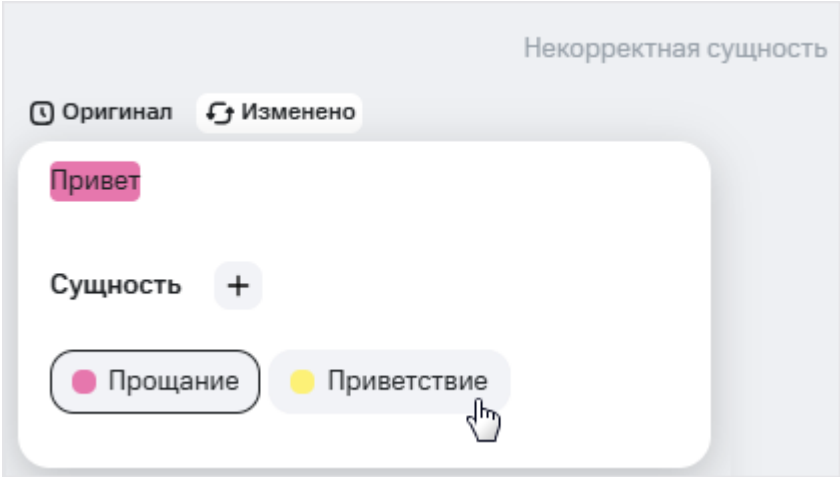
3. Перейдите в режим разметки по кнопке **Разметить**.

1. Выделите шаг и скорректируйте разметку. Под шагом понимается реплика пользователя и ответ бота на нее. Например, если неверно была определена тематика, то выделите шаг, отметьте его соответствующим маркером и выберите нужную тему.



Для разметки используются маркеры:

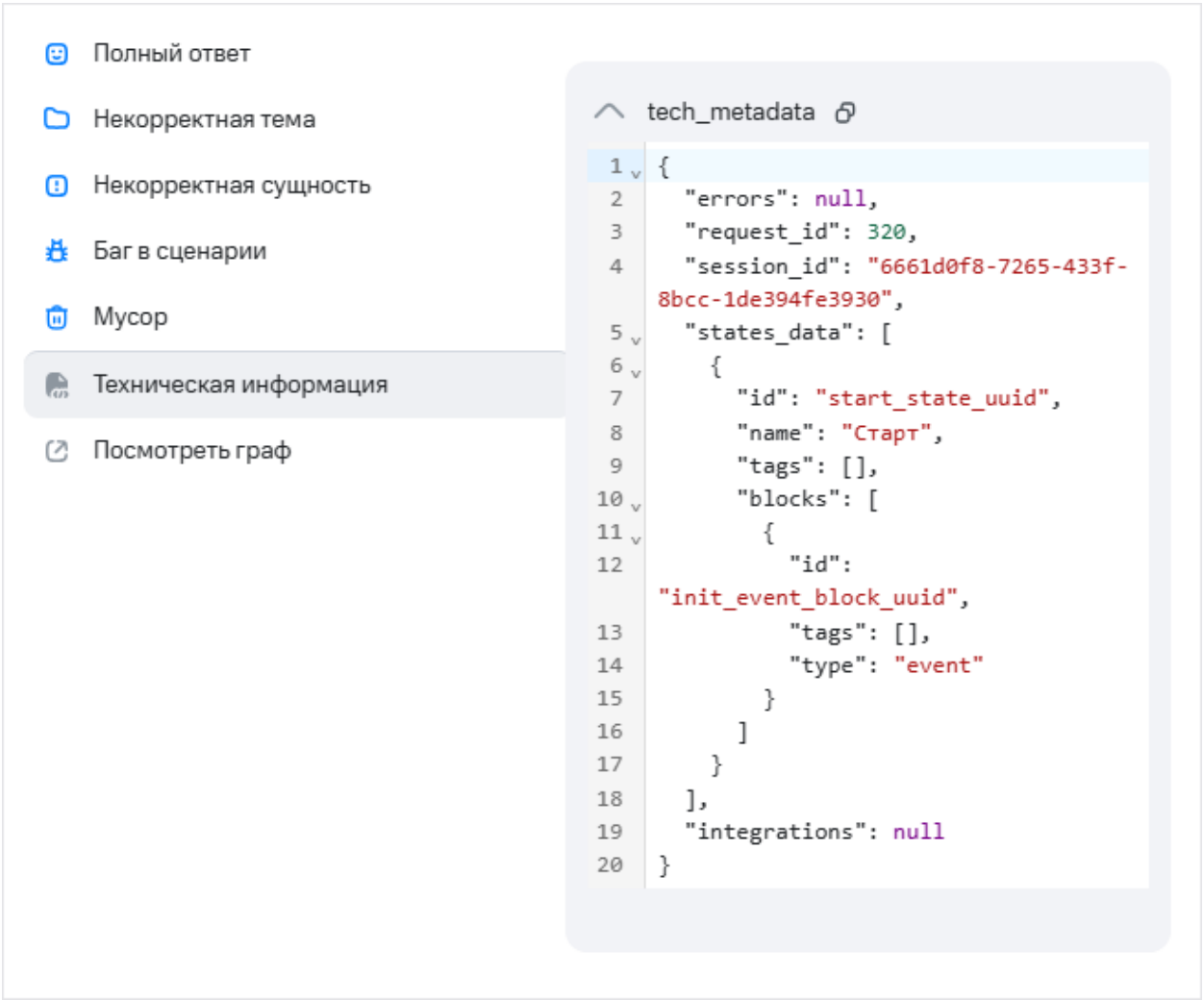
- **Полный ответ**. Выберите этот маркер, если ответ бота удовлетворяет запросу;
- **Некорректная тема**. В этом случае выберите новую тему и оставьте комментарий для следующей разметки;
- **Некорректная сущность**. Выделите слово или его часть и выберите маркер, если сущность не была определена. Если сущность была определена неверно, то предварительно удалите маркер. Для этого наведите на слово и нажмите на ✖ рядом во всплывающей подсказке. Если нужной сущности нет, добавьте ее по кнопке +.



- **Баг в сценарии**. Выберите маркер, если ответ бота не соответствует теме запроса;
- **Мусор**. Отметьте реплику бота, если она избыточна.

Для просмотра вспомогательной информации воспользуйтесь кнопками:

- **Техническая информация**. Выберите пункт, чтобы открыть метаданные диалога в формате JSON. Например, по ней можно определить идентификатор сессии, он указывается в параметре **session_id**. А в параметре **request_id** хранится идентификатор запроса по конкретному шагу диалога.



- **Посмотреть граф**. Нажмите на кнопку, чтобы перейти в сценарий, который был выбран для прохождения на данном шаге.

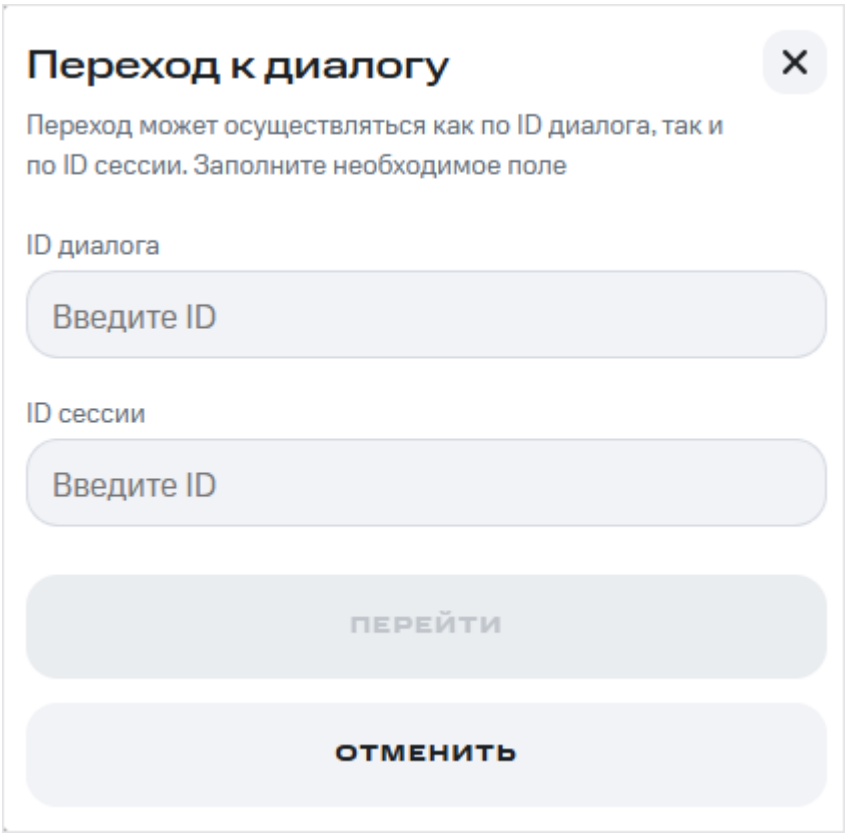
1. При необходимости оцените диалог в целом. Для этого выберите одну из оценок:
   -  – успешный диалог;
   -  – неудачный диалог;
   -  – под вопросом;
   -  – мусор.
2. Завершите работу с разметкой. Для этого нажмите на кнопку **Закончить разметку**.

Поиск диалога

Чтобы перейти к нужному диалогу, вы можете найти его по идентификатору или отфильтровать список по определенным критериям.

Переход по ИД

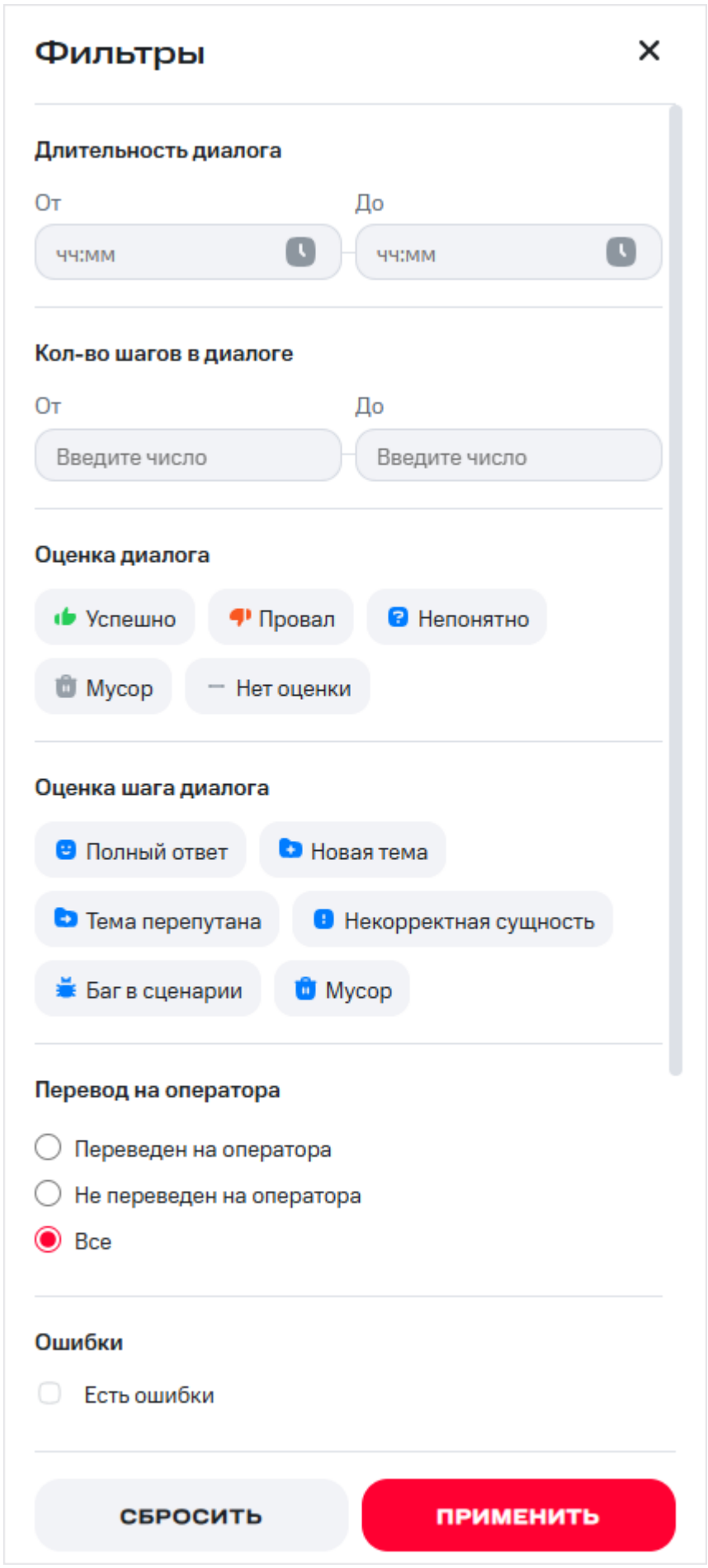
По кнопке **Перейти по ID** открывается модальное окно:



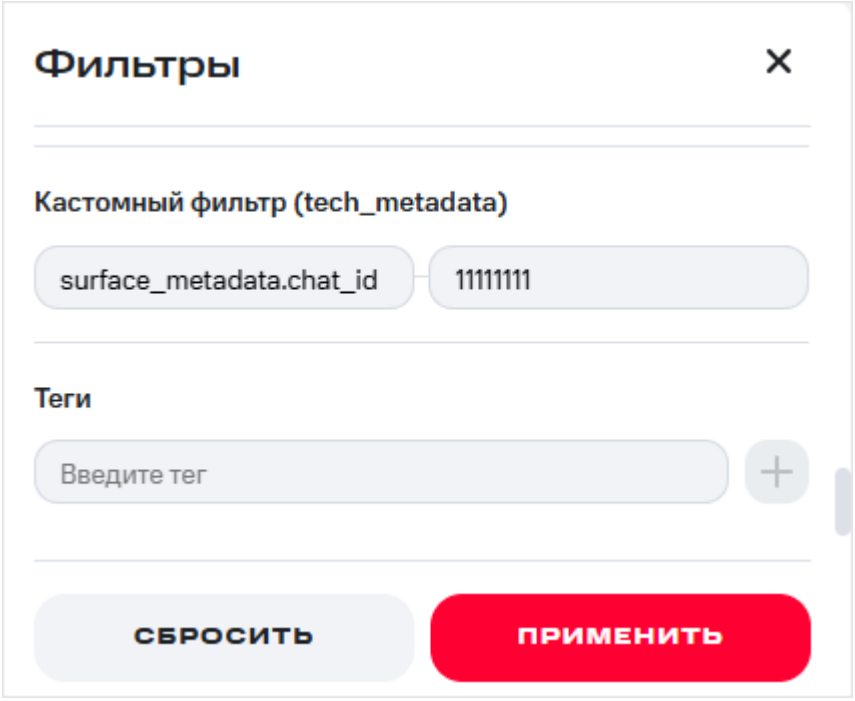
Укажите значение в одном из полей: **ID диалога** или **ID сессии**. Нажмите на кнопку **Перейти**.

Фильтрация диалогов

Вы можете отфильтровать диалоги по длительности, количеству шагов, оценке диалога и т.д. Для этого по кнопке **Фильтры** откройте панель фильтрации и укажите нужные критерии. Примените выбранные фильтры по кнопке **Применить**.


Фильтры
Длительность диалога
От
До
чч:мм
чч:мм
Кол-во шагов в диалоге
От
До
Введите число
Введите число
Оценка диалога
Успешно
Провал
Непонятно
Мусор
Нет оценки
Оценка шага диалога
Полный ответ
Новая тема
Тема перепутана
Некорректная сущность
Баг в сценарии
Мусор
Перевод на оператора
Переведен на оператора
Не переведен на оператора
Все
Ошибки
Есть ошибки
СБРОСИТЬ
ПРИМЕНИТЬ

Если нужно найти диалоги по параметру из JSON проекта, например по идентификатору чата surface_metadata.chat_id, то для этого удобно использовать **Кастомный фильтр (tech_metadata)**:



Обучение классификатора

После разметки экспортируйте размеченные диалоги и привяжите классификатор:

1. Перейдите на страницу **Диалоги**.
2. Чтобы сформировать файл для обучения классификатора, отфильтруйте диалоги. Для этого откройте панель фильтрации по кнопке **Фильтры**, установите нужные признаки и нажмите на кнопку **Применить**. Например, отфильтруйте диалоги по признаку «Тема перепутана».
3. Нажмите на кнопку **Экспортировать**.
4. Перейдите на страницу **Разметка данных**.
5. Откройте созданный ранее проект разметки и загрузите в него сохраненный файл разметки. При загрузке выберите пункт **Сохранять разметку из файла**, чтобы перезаписать существующую разметку. Подробнее о добавлении данных см. в разделе «Добавление данных в проект». Скачайте получившийся датасет.
6. Создайте AI сервис и загрузите в него датасет.
7. Подключите сервис к боту. Для этого перейдите в конструктор нужного бота. В настройках версии в поле **Классификатор** выберите ранее обученный классификатор.

> **ПОДСКАЗКА**
>
> Если в классификаторе появились новые классы, привяжите их к сценариям.

1. Протестируйте бота и опубликуйте в канале.

(пусто)

Использование AI-сервисов

AI-сервисы — используют искусственный интеллект для обработки информации и решения задач в самых разнообразных сферах. Они способны проводить автоматизированный анализ данных, распознавать изображения и речь, выполнять автоматический перевод, предсказательное моделирование, систематизировать неструктурированные данные и многое другое. AI-сервисы платформы AI Agents Platform обеспечивают создание и использование моделей машинного обучения (ML-моделей).

Ключевые преимущества AI-сервисов платформы:

- **No-code подход**: создавайте и обучайте модели без программирования.
- **Масштабируемость**: автоскейлинг и батчинг для обработки пиковых нагрузок.
- **Интеграция**: с LLM (Large Language Models), базами знаний и ботами.
- **Гибкость**: управляйте жизненным циклом сервисов (создание, запуск, остановка, удаление).

Типы AI-сервисов платформы

Платформа поддерживает AutoML-сервисы, включая классификатор и NER.

- **Классификатор** (Classifier) — AutoML-сервис для классификации текстов по классам (например: "запись на прием" или "запрос информации"). Обучается на наборах данных, в которых особое внимание уделено равномерному распределению классов. Это позволяет модели точно определить ключевые намерения пользователя в запросе и направить в нужный сценарий бота.
- **NER** (Named Entity Recognition) — AutoML-сервис для распознавания в тексте именованных сущностей, таких как имена, даты, географические названия и термины. В процессе обучение к датасету NER можно добавить пользовательский словарь синонимов, что повышает точность анализа. Сервис извлекает структурированные данные из неструктурированных текстов, которые затем могут быть использованы в работе ботов.

ТИП СЕРВИСА	ЦЕЛЬ	ФОРМАТ ДАННЫХ	ОБУЧЕНИЕ/ ИНДЕКСАЦИЯ	КЛЮЧЕВЫЕ ОПЦИИ И ОГРАНИЧЕНИЯ
Classifier	Категоризация по классам	CSV (text, label; мин. 2 класса, 2 примера на класс)	Fine-tuning, SetFit, AncSetFit	необходим баланс классов; опциональный anc_label для малых датасетов; для качества минимально 81 пример на класс
NER	Распознавание сущностей	JSON (с entities; до 100 Мб)	Fine-tuning, SetFit с словарями синонимов (JSON)	словарь загружается после датасета; улучшает распознавание синонимов; извлекает структурированную информацию

Создание и использование AI-сервиса

AI-сервис создается в статусе **Черновик**. Для любого типа сервиса следуйте приведенным ниже шагам. Вы можете прервать процесс и вернуться позже — сервис сохранится.

1. Перейдите в раздел **AI-сервисы** и нажмите **Создать сервис**.

2. Выберите тип сервиса.
3. Задайте название сервиса.

> **ПРИМЕЧАНИЕ**
>
> Название генерируется автоматически в формате "draft-<uuid>". Для удобства рекомендуется заменить его на свое. Дубликаты названий подсвечиваются предупреждением «Данное имя уже занято».

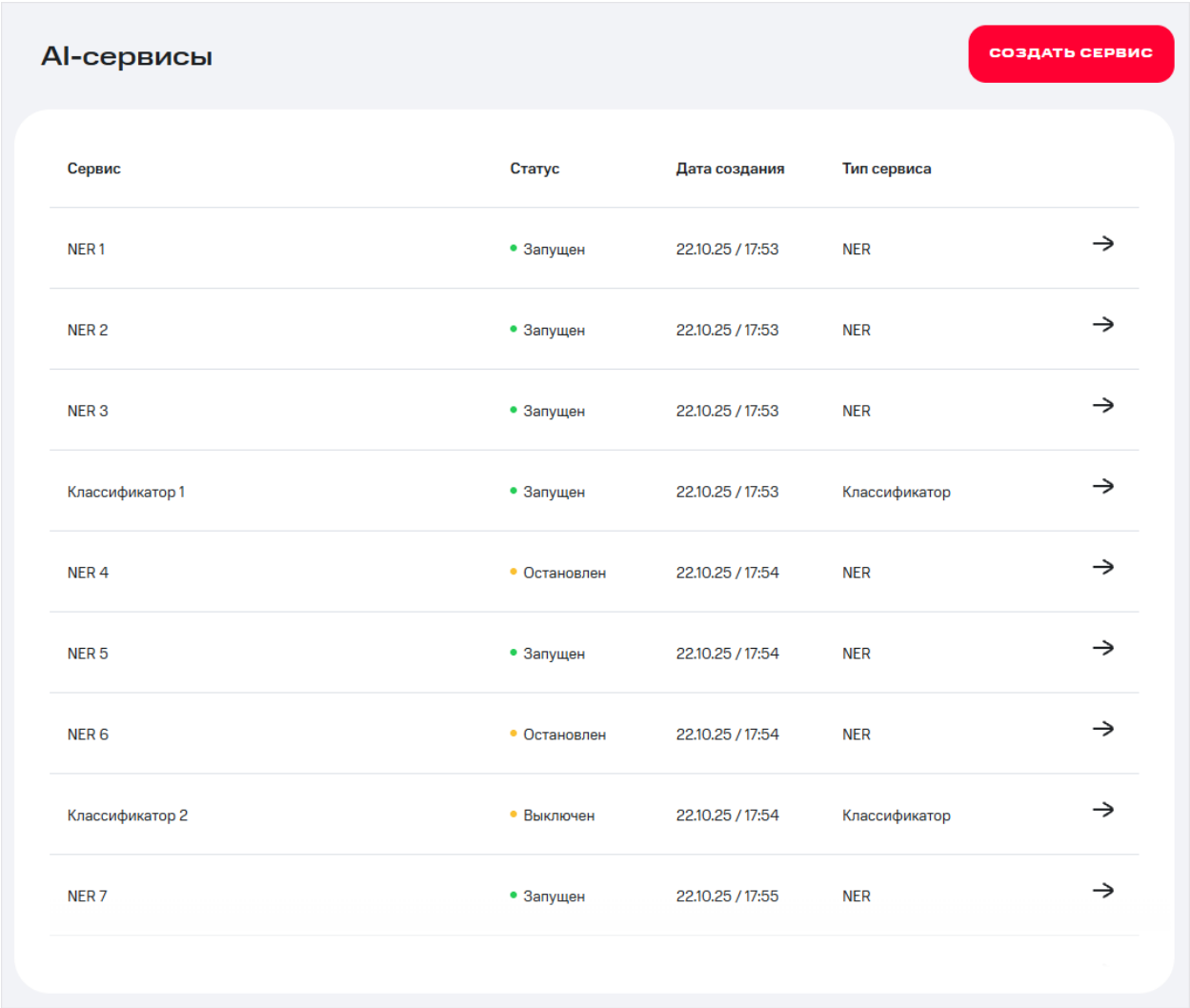
4. Задайте параметры сервиса. Подробнее см. разделы «Настройка AutoML-сервиса».
5. Загрузите данные: датасет или документы для базы знаний.
6. Запустите обучение/индексацию.
7. Протестируйте сервис в виджете.
8. Подключите сервис к проекту. Подробнее см. разделы «Подключение обученной модели к боту».
9. Запустите сервис с помощью переключателя. При необходимости сервис можно остановить или удалить.

> **ВНИМАНИЕ**
>
> Удаление сервиса необратимо. Удаляются модель, датасет из S3-хранилища и записи в базе данных.

В карточке сервиса отображается его статус:

- **Черновик**
- **Создаётся**
- **Запускается** (только Классификатор и NER)
- **Запущен**
- **Выключен**
- **Удаляется**
- **Ошибка**

Элементы интерфейса для управления AI-сервисами



Описание пользовательского интерфейса представлено в таблице.

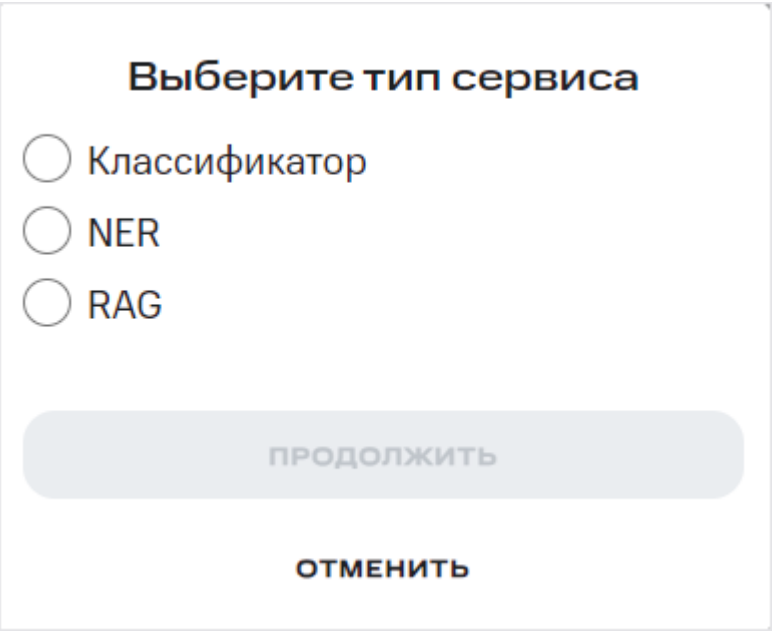
ЭЛЕМЕНТ	ОПИСАНИЕ	РАЗЛИЧИЯ В ЗАВИСИМОСТИ ОТ ТИПА СЕРВИСА И СОВЕТЫ
Список сервисов	Таблица с именем, статусом, датой создания, типом сервисов, кнопка **Создать сервис**	- общий вид для всех AI-сервисов; - если таблица пуста, значит сервисы еще не созданы или все удалены; - кликните строку таблицы для перехода в карточку AI-сервиса
Создание сервиса	Здесь выполняется базовая настройка и загрузка данных сервиса	- для NER – поле для словаря; - скачайте шаблон датасета для образца; - проверьте форматирование датасета перед загрузкой; - отмените создание сервиса для удаления черновика

ЭЛЕМЕНТ	ОПИСАНИЕ	РАЗЛИЧИЯ В ЗАВИСИМОСТИ ОТ ТИПА СЕРВИСА И СОВЕТЫ
Карточка сервиса	Полная информация о сервисе на вкладках **Детали** и **Настройки**	- для NER – словарь; - настройки AutoML-сервисов меняются только в остановленном состоянии; - сохраните изменения для применения
Тестирование	Виджет для ввода запроса к сервису и просмотра ответа в формате JSON	- Классификатор: label/score; - NER: labels (text, label, start/end, score); - доступно только для запущенных сервисов

Создание нового AutoML-сервиса

Чтобы создать новый AutoML-сервис:

1. Перейдите в раздел **AI-сервисы** и нажмите кнопку **Создать сервис**.



2. Выберите тип AutoML-сервиса.
3. Нажмите **Продолжить**.
4. В окне **Создать сервис** в поле **Название сервиса** автоматически сформировано название, состоящее из стартового статуса сервиса – draft (черновик) и его uuid – 36-символьного универсального уникального идентификатора. Измените стартовое название. При необходимости вы сможете сделать это позднее в карточке сервиса.

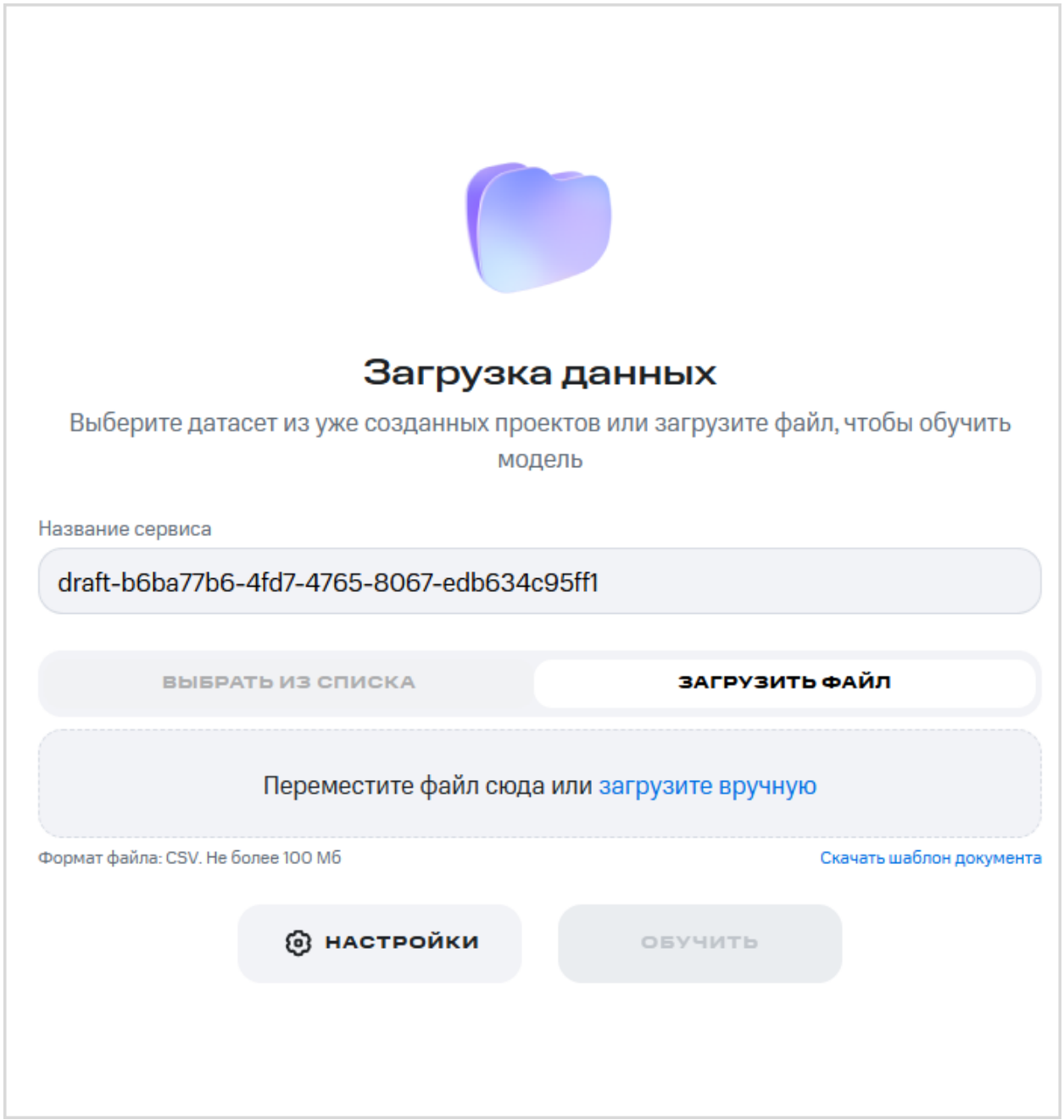
В результате AutoML-сервис создан и готов к настройке и загрузке данных для обучения модели.

Различия AutoML-сервисов

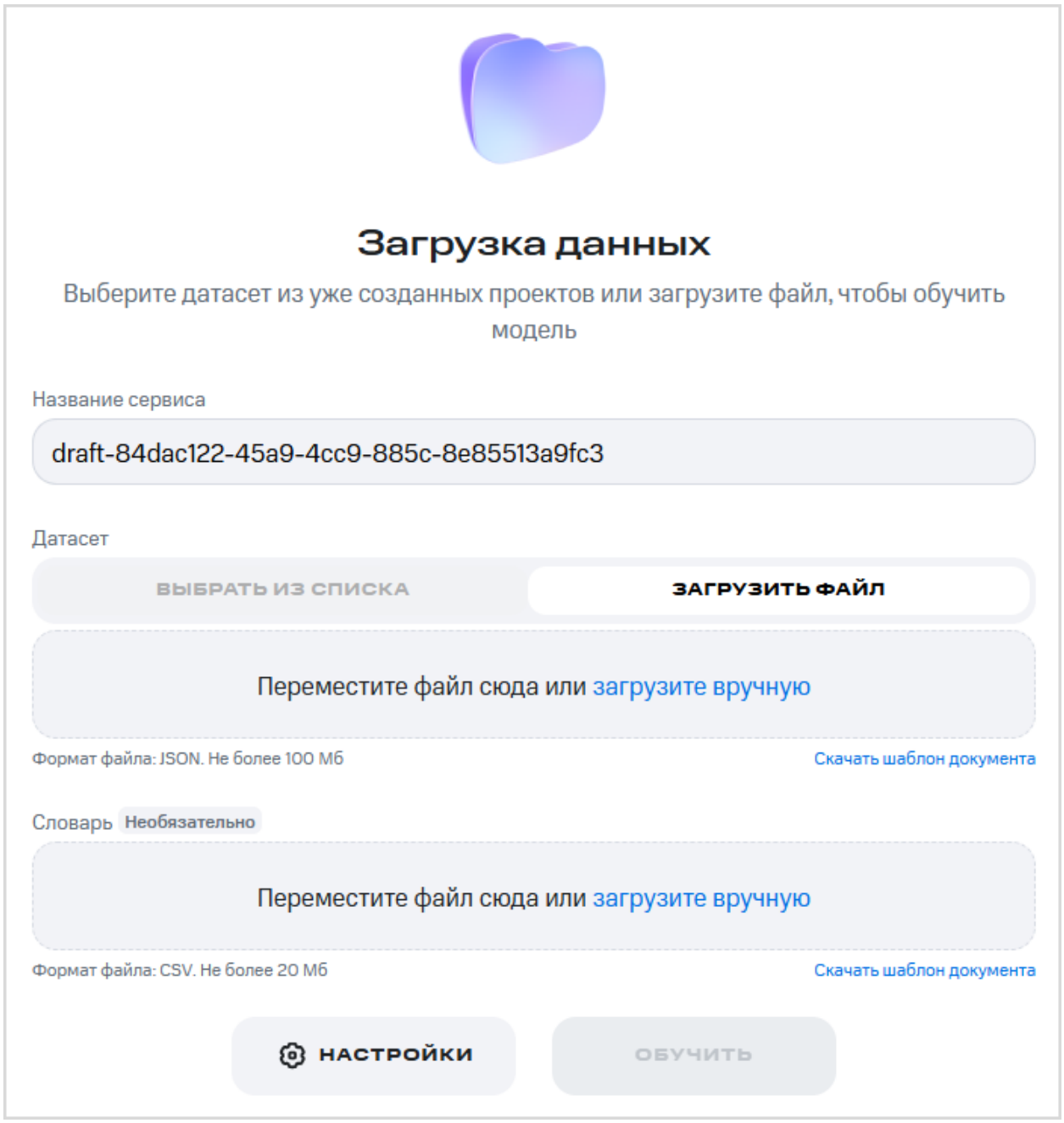
АСПЕКТ	КЛАССИФИКАТОР	NER
Цель модели	Распределение по классам	Распознавание сущностей
Формат датасета	CSV	JSON

АСПЕКТ	КЛАССИФИКАТОР	NER
Дополнительно данные для загрузки	Нет	Словарь синонимов CSV
В карточке	Классы	Сущности, просмотр словаря, F1-мера, confusion matrix
Тестирование	Top-N классов, score	Top-N классов, score

Окно создания сервиса типа **Классификатор**:



Окно создания сервиса типа **NER**:



Настройка AutoML-сервиса

ML-модели требуют значительных вычислительных ресурсов. Если нагрузка пиковая, система может «задохнуться»: задержки вырастут, запросы начнут падать, серверы с развернутыми моделями могут выйти из строя.

Чтобы оптимизировать нагрузку, в AI Agents Platform реализованы стратегии автоскейлинга и батчинга.

- **Автоскейлинг** – это автоматическое горизонтальное масштабирование, при котором система в зависимости от нагрузки динамически добавляет или удаляет реплик (копии) ML-моделей. Автоскейлинг отслеживает метрики определенных ресурсов системы, например CPU/GPU или памяти. При достижении метриками установленных значений автоматически запускаются дополнительные реплики моделей. Когда нагрузка падает, лишние реплики удаляются, чтобы не тратить ресурсы зря.
- **Батчинг** – это метод обработки данных, при котором несколько входящих запросов группируются в один "батч" (пакет) и обрабатываются моделью одновременно. Вместо обработки запросов по

одному, модель получает их пачкой, что оптимизирует использование аппаратных ресурсов. Батчинг увеличивает количество обработанных запросов в единицу времени. Запросы накапливаются в буфере до достижения заданного размера батча, длительности задержки или таймаута, чтобы не задерживать слишком долго.

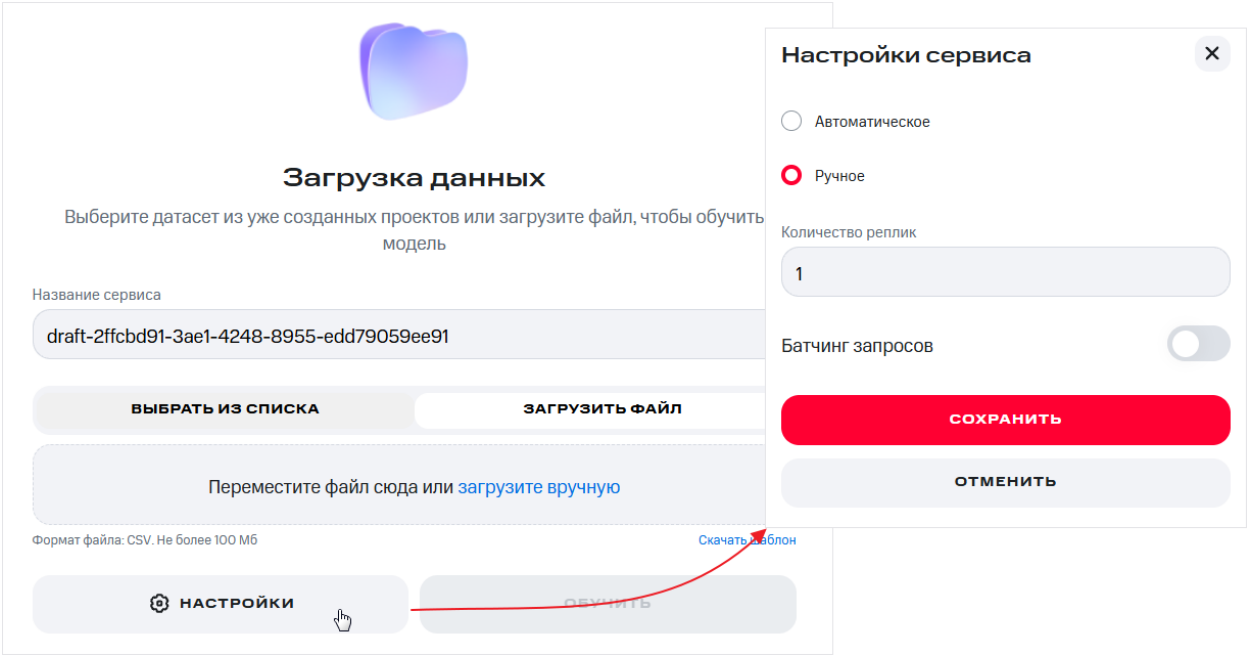
Если автоскейлинг и батчинг «работают» вместе, то при пиковом увеличении количества запросов батчинг группирует их в пакеты, а если при этом возрастет нагрузка на аппаратные ресурсы – автоскейлинг запустит дополнительные реплики модели.

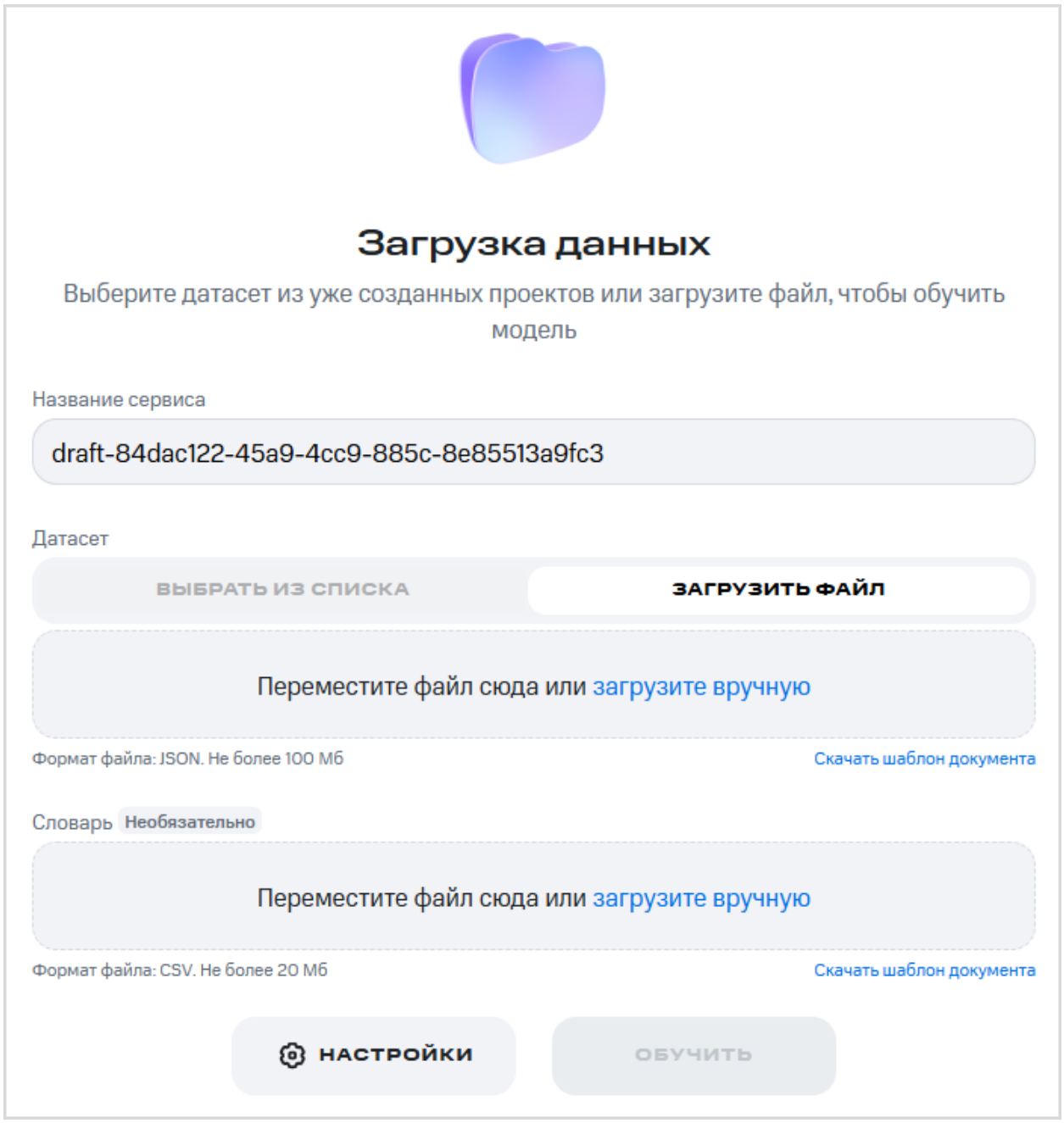
Платформа поддерживает ручной режим скейлинга, когда количество инстансов модели задается при создании AutoML-сервиса и балансировка выполняется между ними без репликации.

По умолчанию новый AutoML-сервис создается со скейлингом, установленным в ручном режиме, и отключенным батчингом запросов.

Чтобы изменить настройки:

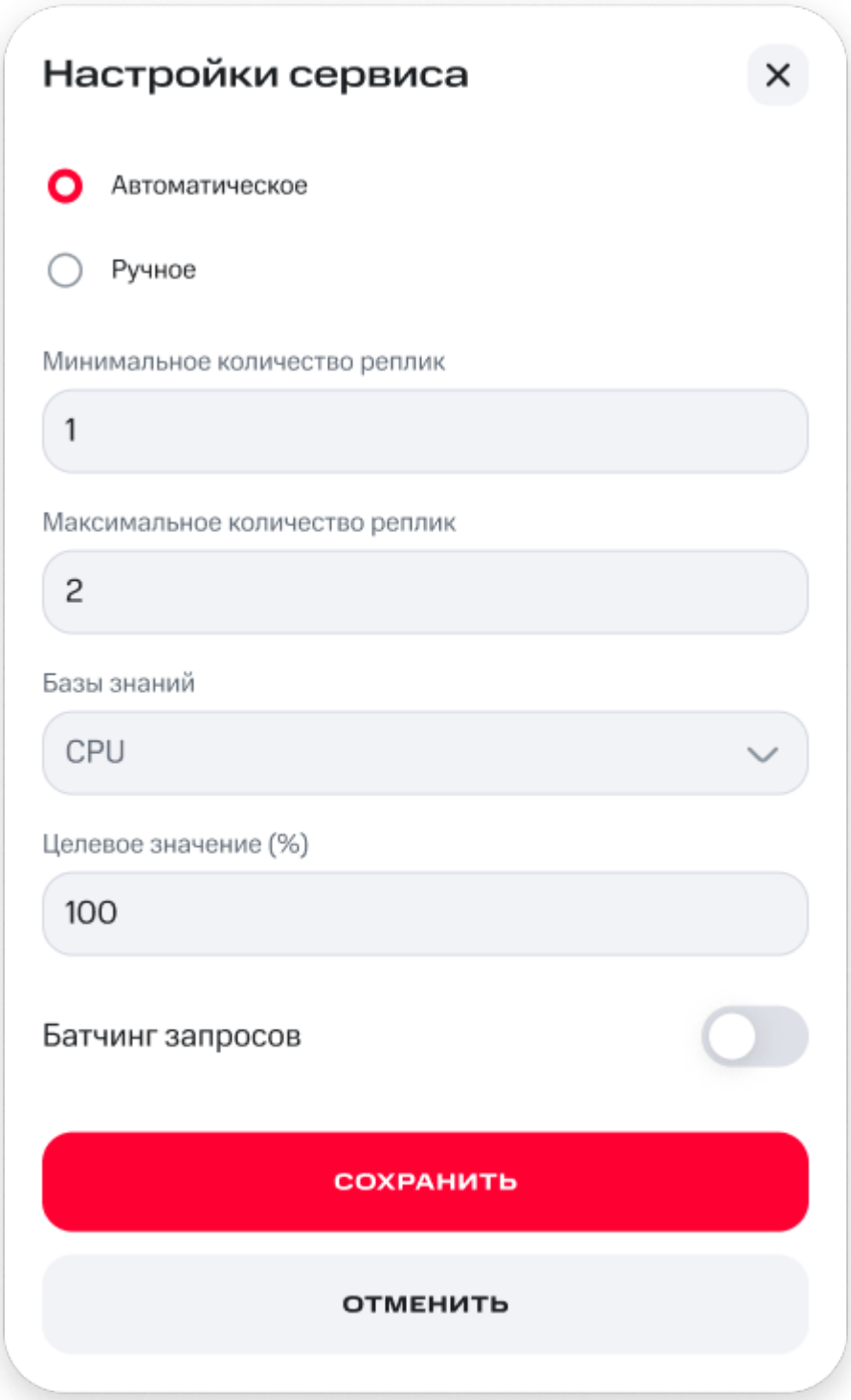
1. На странице **Создать сервис**, нажмите кнопку **Настройка** – откроется диалоговое окно **Настройка сервиса**.





Если скейлинг остается в ручном режиме, то установите необходимое количество реплик. Минимальное значение 1, верхний предел не ограничен. Но стоит учитывать возможности ваших ресурсов: большое число реплик может негативно повлиять на систему.

1. Если нужно включить автоскейлинг, поставьте отметку **Автоматическое** – в диалоговом окне появятся поля для его настройки.
    - Установите минимальное и максимальное количество реплик.
    - Выберите метрику значение которой будет отслеживаться – CPU или память.
    - Установите целевое значение нагрузки от 1 % до 100 %.



3. Если необходимо включить батчинг, переведите переключатель **Батчинг запросов** вправо — появятся поля для настройки.

- Установите максимальный размер батча, т.е. максимальное количество запросов, которое может быть сгруппировано в батч для одновременной обработки моделью. Чем более высоконагруженной является ваша система, тем выше может быть значение этого параметра. При этом большой размер батча требует больше ресурсов памяти.
- Установите значение времени максимальной задержки в миллисекундах. В течение этого времени система будет ожидать накопления запросов в батче, прежде чем отправить его на обработку. Если

во время ожидания задержка достигнет установленного значения, то батч отправится в ML-модель даже если он не полный. Это предотвращает бесконечное ожидание при низкой нагрузке.

- Установите таймаут обработки — максимальное время в секундах, отведенное на обработку одного батча или запроса моделью. Если длительность обработки превысит заданную, то запрос будет считается неудачным, а система может повторить или отклонить его. Чем сложнее задача в боте с ML-моделью (например, анализ длинного текста или генерация идей), тем больше времени нужно на обработку — устанавливайте длительный таймаут. Иначе бот может зависнуть или отклонить запрос, чтобы не тратить ресурсы зря. Для простых вопросов хватит 10 секунд, а для тяжёлых — до минуты.

> **ПОДСКАЗКА**
>
> Для высокой нагрузки: большой размер батча + умеренная задержка = высокое количество обработанных запросов в единицу времени. Для низкой задержки ответов: маленький размер батча + маленькая задержка = быстрые ответы, но ниже эффективность.

1. Сохраните настройки.

Сервис настроен. Изменить значения параметров можно в соответствующем разделе карточки AutoML-сервиса.

Загрузка данных для обучения ML-модели сервиса

Загрузить данные можно двумя способами:

- Выбрать существующий проект разметки данных. Подробнее о разметке см. раздел «Разметка данных»;
- Самостоятельно создать и разметить датасет, затем загрузить его в AutoML-сервис файлом со своего компьютера.

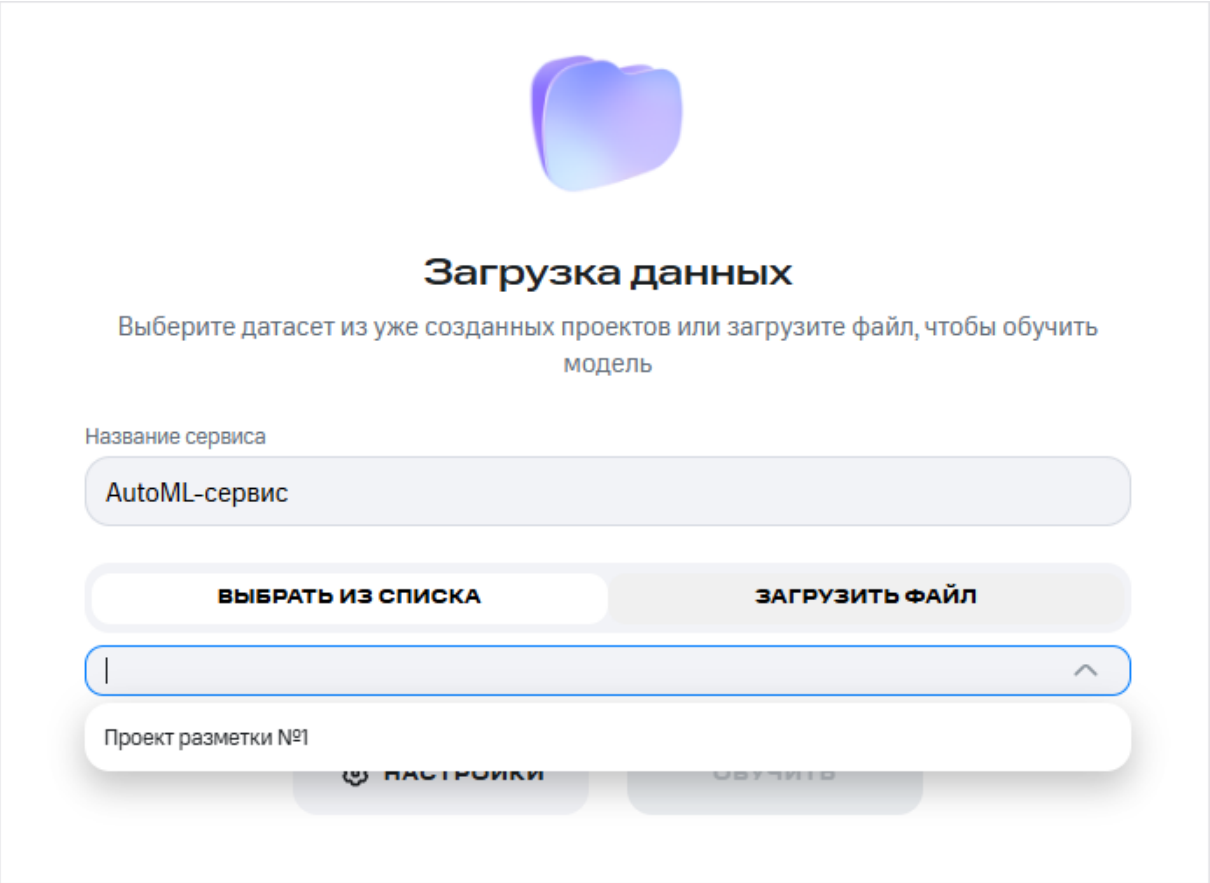
Выбор для обучения проекта разметки данных

Чтобы выбрать для обучения ML-модели готовый проект разметки:

1. Нажмите **Выбрать из списка** — появится поле выбора проекта разметки.

> **ВНИМАНИЕ**
>
> Если в системе еще не существует ни одного проекта разметки для обучаемого сервиса, кнопка Выбрать из списка будет неактивна. Для ее активации необходимо создать хотя бы один проект разметки (подробнее см. здесь Создание проекта разметки данных).Если в системе еще не существует ни одного проекта разметки для обучаемого сервиса, кнопка Выбрать из списка будет неактивна. Для ее активации необходимо создать хотя бы один проект разметки. Подробнее см. раздел «Создание проекта разметки данных».

2. Выберите проект в выпадающем списке:



3. Активируется кнопка **Обучить**.

Данные загружены в сервис. Перейдите к обучению.

Загрузка файла датасета

В качестве датасета для обучения моделей используются:

- Для классификатора – csv-файл размером не более 100 Мб. Информация о создании качественных датасетов для классификатора представлены в разделе Рекомендации по созданию датасетов для обучения классификатора.
- Для NER – JSON. Не более 100 Мб.

Чтобы подготовить и загрузить файл датасета:

1. На странице **Создать сервис** нажмите **Скачать шаблон документа**. На ваш компьютер загрузится заранее заполненный и правильно размеченный датасет с тестовыми данными в формате, соответствующем типу сервиса, который вы создаете (csv или json). Работайте прямо в скачанном шаблоне, затем сохраните его с нужным именем.
2. Готовый файл можно перетащить мышкой в область загрузки или щелкнуть в области загрузки и загрузить через проводник операционной системы. На экране вы увидите загружаемый файл и индикатор загрузки, который при успешном выполнении процесса изменится на иконку ✕. При необходимости загрузку можно отменить.

> **ВНИМАНИЕ**
>
> В процессе загрузки датасета система проверяет его формат (расширение файла) и размер. Если файл не прошел проверку – вы увидите сообщение **Неправильный формат файла** или **Превышен максимальный размер файла**. Кнопка Обучить останется неактивной. Решение – смена формата на требуемый или уменьшение размера файла.

3. После того, как датасет успешно загрузится, кнопка **Обучить** станет активной. При необходимости датасет можно удалить.

> **ПРИМЕЧАНИЕ**
>
> На этом шаге создания сервиса вы можете перейти в другой раздел веб-интерфейса платформы или выйти из системы. Создаваемый вами сервис сохранится как черновик и вы сможете продолжить работу с ним позже. Файла датасета останется прикрепленным. Обратите внимание, если вы уже успели переименовать сервис при создании, то его название вернется к изначальному системному формату – draft– uuid

Файл датасета загружен. Перейдите к обучению.

Загрузка пользовательского словаря синонимов сущностей для NER

Загрузка словаря – это не обязательное действие. При этом словарь улучшает качество домен-специфичных моделей NER, обученных на сущностях из какой-либо одной отрасли, например медицины. Модель «обогащенная» словарем учитывает синонимы из него в своих ответах, что может быть полезно для ботов с вариативным вводом (когда клиент отправляет в запросе, например, «ЛОР» вместо «Оториноларинголог»).

Формат словаря синонимов сущностей – это CSV-таблица, в которой для каждого типа определенной в датасете NER сущности (label) хранятся:

- Нормализованная форма (каноническое название сущности);
- Варианты написания (синонимы сущности, среди которых будет выполняться поиск и нормализация).

> **ВНИМАНИЕ**
>
> Словарь – неотделимая часть сервиса и загружается только при создании. Удалить словарь из сервиса нельзя. Чтобы модель перестала использовать словарь в ответах, вам придется пересоздать сервис без словаря.

Чтобы создать и загрузить словарь:

1. Создайте NER-сервис. Загрузите в него датасет. Это обязательный шаг – без датасета на одном словаре обучить NER не возможно.
2. В окне **Создание сервиса** скачайте шаблон словаря.
3. Работайте со словарем прямо в скачанном файле. Заполните его в соответствии с форматом (см. ниже).

4. Готовый словарь загрузите в поле загрузки окне **Создание сервиса**. Система проверит файл. Если выявится несоответствие правилам валидации, отобразится сообщение рядом с полем загрузки словаря. Кнопка обучение станет активна.

NER-сервис готов к обучению с интеграцией пользовательского словаря синонимов.

Формат csv-файла словаря:

```
label,canonical_name,aliases
исследование,МРТ головного
мозга,магнитно-резонансная томография головного мозга\tМРТ ГМ
исследование,КТ брюшной полости,компьютерная томография брюшной полости\tКТ БП
лекарство,ацетилсалициловая кислота,аспирин\tАСК\tацетилсалициловая кислота
симптом,головная боль,цефалгия\tголовная боль\tболь в голове
```

Описание полей csv-файла:

КОЛОНКА	ОПИСАНИЕ	ПРИМЕР
label	Тип сущности из датасета NER	исследование, лекарство, симптом
canonical_name	Нормализованная (каноническая) форма	МРТ головного мозга, ацетилсалициловая кислота, головная боль
aliases	Варианты написания (синонимы через \t)	магнитно-резонансная томография головного мозга\tМРТ ГМ

Управление словарем

ДЕЙСТВИЕ	ПОРЯДОК ВЫПОЛНЕНИЯ	РЕЗУЛЬТАТ
Загрузка словаря	Только при создании сервиса	Выбрать CSV-файл и загрузить
Просмотр статуса словаря	AI-сервисы → Карточка сервиса → Детали сервиса	Отображается поле **Словарь**
Скачивание словаря	AI-сервисы → Карточка сервиса → Детали сервиса	AI-сервисы → Карточка сервиса → Детали сервиса
Удаление словаря	Невозможно	Удалить словарь нельзя. Только пересоздать сервис
Замена словаря	Невозможно	Заменить словарь нельзя. Только пересоздать сервис

Ограничения для словарей

ТИП ОГРАНИЧЕНИЯ	ОПИСАНИЕ
Размер файла	Максимальный размер – 20 Мб

ТИП ОГРАНИЧЕНИЯ	ОПИСАНИЕ
Формат	Только CSV
Типы сущностей	Только из датасета, на котором будет обучен NER
Количество словарей	1 словарь на сервис

Также нельзя выполнить следующие действия:

- загрузить словарь после создания сервиса;
- создать NER-сервис только со словарем без датасета;
- удалить словарь из NER-сервиса;
- заменить словарь в обученном сервисе;
- использовать в словаре только сущности, которых нет в датасете;
- использовать в словаре некоторые сущности, которых нет в датасете.

Обучение ML-модели

При загрузке датасета система автоматически выбирает оптимальный метод обучения. Выбор зависит от количества примеров в датасете и их распределения по классам. Система всегда стремится выбрать метод, который даст наилучший результат именно для ваших данных.

При обучении используются следующие методы:

Fine-tuning (Дообучение) – мощная предобученная языковая модель адаптируется под конкретную задачу. К модели добавляется новый слой для категорий, представленных в датасете, после чего вся модель или её часть дообучается на этих данных. Fine-tuning выбирается в качестве способа обучения для датасетов с 81+ примеров на каждый класс и хорошим балансом данных. Обучение может занимать длительное время. При достаточном объёме данных достигается максимальная точность обученной модели.

SetFit – модель учится различать тексты, сравнивая их между собой – какие похожи (из одного класса), а какие отличаются (из разных классов). После этого обучается компактный классификатор, который работает с векторными представлениями текстов. Способ обучения определяется для датасетов, содержащих от 8 до 80 примеров на класс. Обучение происходит быстрее,чем Fine-tuning. В итоге даже на малых данных получается хорошее качество модели.

AncSetFit (Anchored SetFit) – усовершенствованная версия SetFit, которая автоматически находит в датасете наиболее типичные примеры для каждого класса (якоря) и использует их как эталоны при обучении. В нашей системе якоря указываются в столбце anclable на русском языке. Это помогает модели лучше понять суть каждой категории. Способ обучения задействуется, если в датасете от 2 до 5 классов. Лучшие результаты обучения на минимальном количестве данных.

Чтобы начать обучение модели, нажмите кнопку **Обучить**. На экране появится сообщение о том, что модель на обучении.

ПОДСКАЗКА

Обучение модели занимает некоторое время. Пока процесс идет, ничто не мешает вам продолжить работу в других разделах AI Agents Platform. При этом вы можете периодически проверять статус вашего сервиса в разделе AI-сервисы. В ходе процесса система выполняет проверку качества датасета. Если эта проверка занимает длительное время вы можете увидеть сообщение Проверка данных. Но обычно проверка выполняется быстро и это сообщение не появляется.

При необходимости вы можете отменить создание сервиса в процессе обучения. Отмена подразумевает полное удаление сервиса.

Если при обучении возникает ошибка, на экране отображается соответствующее сообщение. На странице **Список сервисов** сервис, получивший ошибку в процессе обучения, приобретает статус **Ошибка**. Чтобы исправить ситуацию:

- Нажмите на кнопку **Попробовать снова**.
- Если ситуация не изменилась – самостоятельно проверьте датасет на ошибки. Если ошибки найдены – нажмите кнопку Отменить создание, подтвердите удаление сервиса и попробуйте создать новый, используя исправленный файл датасета.
- Если ошибки не нашлось или ситуация повторяется и с новым датасетом – обращайтесь по указанным в сообщении контактам.

Когда обучение успешно завершится, откроется карточка сервиса и начнется процесс его запуска, т.е. развертывания готовой ML-модели в рабочей среде. Статус сервиса в списке изменится на **Запускается**.

ПРИМЕЧАНИЕ

Если во время запуска модели открыть карточку сервиса, в графе **Результаты** обучения будет отображаться положительный статус, что обучение модели успешно завершено. Ход запуска модели отображается в графе **Статус**. Отключить сервис во время запуска нельзя, переключатель заблокирован.

Остановка обучения в ходе проверки данных для переразметки

Если по результатам проверки данных будет определено, что датасет недостаточно качественный, процесс обучения модели остановится.

Проверка данных

Проверка данных завершена. Если вы хотите внести изменения, обновите данные и дождитесь окончания обучения модели.

ОБНОВИТЬ ДАННЫЕ | ПРОДОЛЖИТЬ ОБУЧЕНИЕ

ВНИМАНИЕ

Смысл этой остановки в том, чтобы предоставить пользователю возможность обновить данные, т.е. переразметить датасет. Подробнее см. раздел «Разметка данных». А затем продолжить обучение модели на обновленном датасете.

При этом остается возможность продолжить обучение и без переразметки, на текущем датасете. Для этого нажмите **Продолжить обучение**.

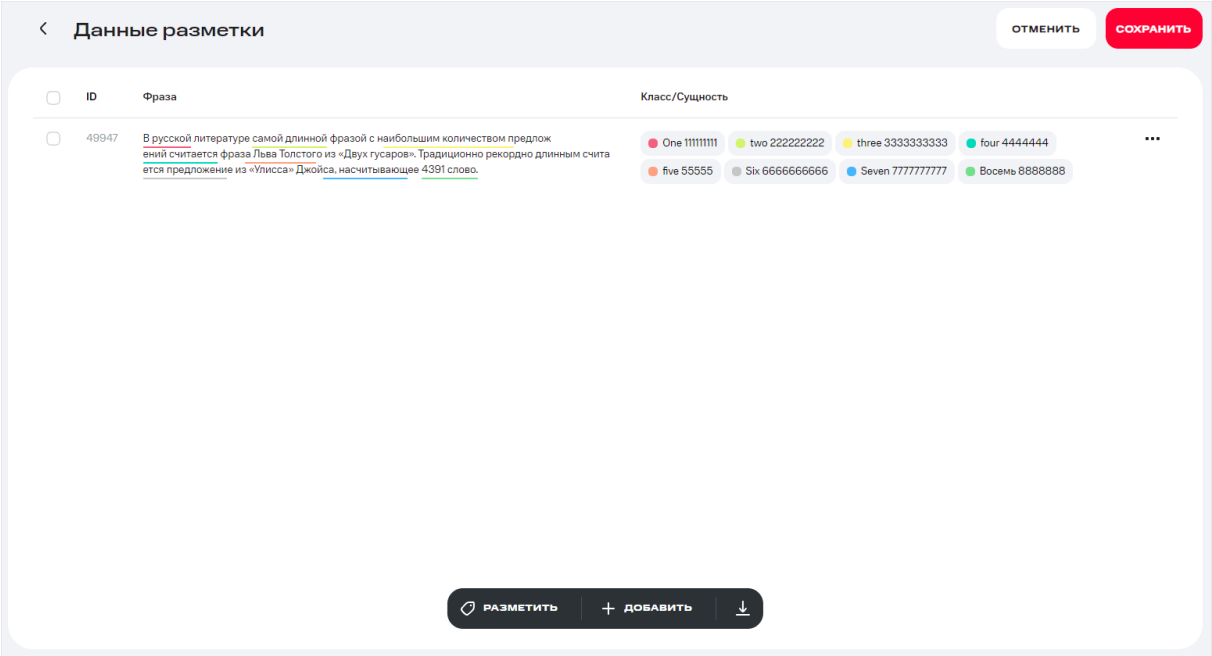
Чтобы обновить данные:

1. Нажмите **Обновить данные** – откроется страница **Данные разметки**. На странице в виде списка представлены те данные, которые в ходе проверки были определены как некачественные.

ПОДСКАЗКА

Чтобы пользователь мог работать именно с некачественными данными, в системе выполняются следующие действия:

- Из датасета, который был загружен в AutoML-сервис при его создании, удаляются валидные данные полученные по результатам проверки. Для оставшихся данных генерируются метки (лейблы).
- По шаблону создается новый проект разметки, основанный на данных и лейблах первого шага.
- Создается копия этого проекта и именно в ней работает пользователь.



2. Внесите изменения. Подробнее о разметке данных см. в разделе «Разметка данных».

> **ВНИМАНИЕ**
>
> Обратите внимание, что внесенные изменения нельзя отменить. Если вы ошиблись и нужно вернуться в первоначальное состояние вариант только один:
>
> - Нажмите **Отменить** – вы окажетесь на экране **Проверка данных**.
> - Нажмите **Обновить данные** – откроется страница **Данные разметки**.

3. Нажмите кнопку **Сохранить** – откроется страница **Данные разметки**.
4. Нажмите **Продолжить обучение**.

В результате обучение продолжится на исправленных данных

Карточка AutoML-сервиса

Карточка сервиса содержит детальную информацию о сервисе и одновременно выполняет роль «пульта управления».

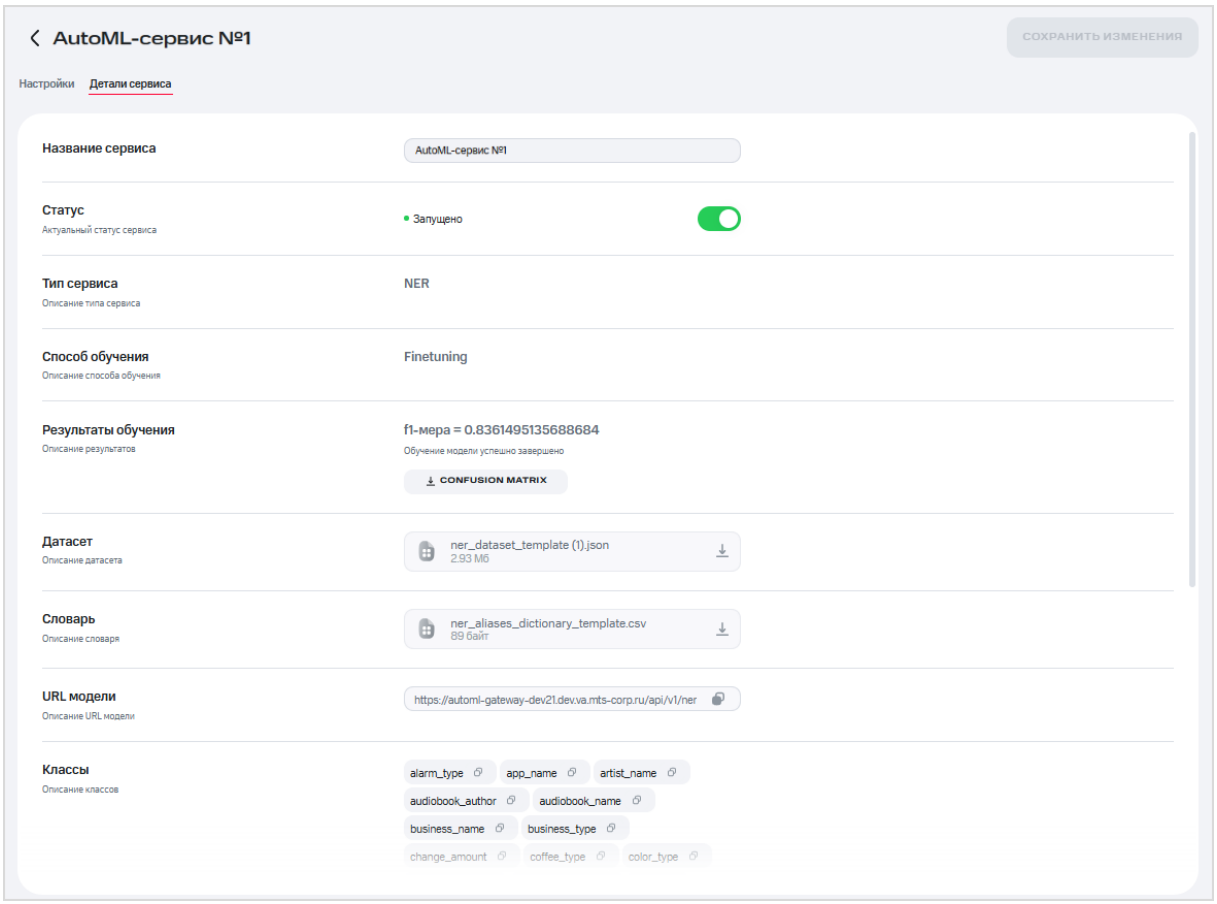
Все изменения сделанные в карточке AutoML-сервиса вступают в силу только после нажатия кнопки **Сохранить изменения**.

Детали сервиса

На вкладке вы найдете:

- **Название сервиса** – доступно для изменения.
- **Статус** – текущий статус сервиса и переключатель, запускающий или останавливающий работу сервиса.
- **Тип сервиса** – Классификатор или NER.
- **Способ обучения** – наименование метода, которым была обучена модель. Подробнее см. раздел «Обучение ML-модели».
- **Результаты обучения** – данные о результатах обучения. Если сервис обучен по методу Fine-tuning, то отображаются f1-мера и confusion matrix.

- **Датасет** – указан файл датасета с возможностью скачать его. Если датасет был некачественным, дополнительно можно скачать отчет о его проверке в формате csv.
- **Словарь** (только для NER) – если словарь загружен – отображается название файла с возможностью скачать его. Если NER обучен без словаря – поле отсутствует в карточке.
- **URL сервиса** – адрес модели, который можно скопировать и использовать для взаимодействия с моделью как ботов, созданных в AI Agents Platform (см. раздел «Проекты»), так и сторонних приложений.
- **Классы** – теги классов или сущностей (для NER), которые модель может определить после обучения. Тег можно копировать.



Настройки

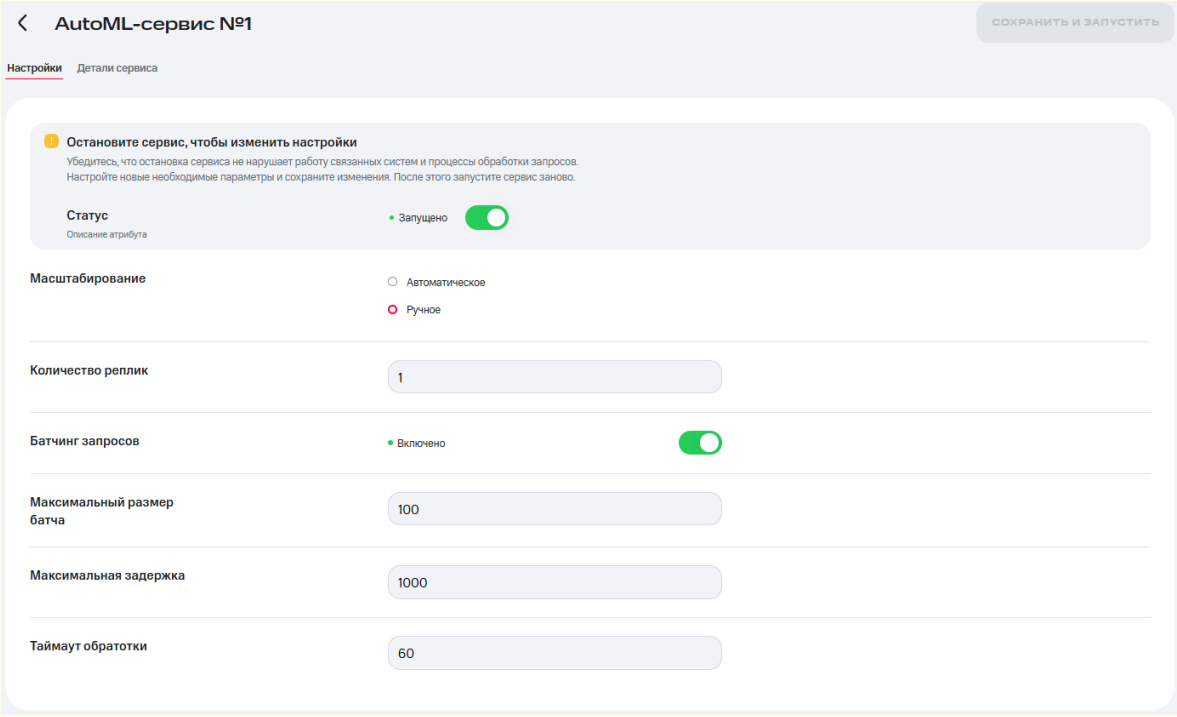
На вкладке можно изменить параметры автоскейлинга и батчинга, заданные во время создания сервиса. Подробнее см. раздел «Настройка AutoML-сервиса».

ВНИМАНИЕ

Редактировать параметры можно только у остановленного AutoML-сервиса. Пока сервис запущен, кнопка **Сохранить и запустить** неактивна.

Прежде чем остановить сервис, обязательно убедитесь, что его остановка не повлияет на работу связанных систем, так как модель сервиса может использоваться ботом.

После настройки сервиса нажмите **Сохранить и запустить**.

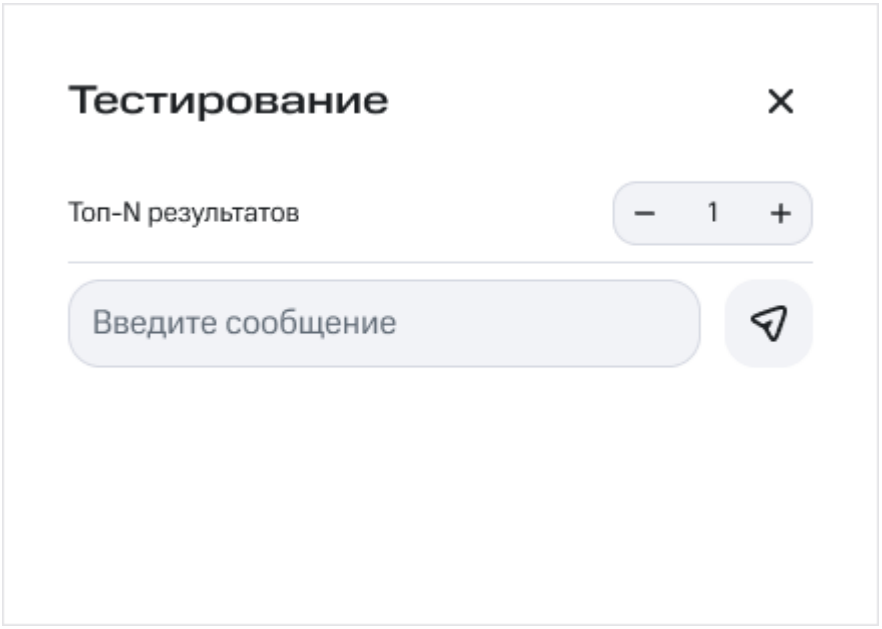


Тестирование AutoML-сервиса

Тестирование AutoML-сервиса необходимо для проверки качества ответов модели и правильности разметки данных, на которых она обучена.

Чтобы протестировать AutoML-сервис:

1. В списке **AI-сервисы** найдите сервис, тестирование которого нужно провести. Перейдите в карточку сервиса.
2. Убедитесь, что сервис запущен. Если нет – запустите сервис переключателем , затем нажмите кнопку Сохранить изменения. Дождитесь пока сервис запустится.
3. Нажмите кнопку **Тестировать** – появится виджет тестирования.
4. При необходимости воспользуйтесь функцией **Top-N**. Для классификатора с ее помощью задается количество гипотез в ответе, отсортированных в порядке убывания параметра **score** (величины, которая показывает насколько текст запроса близок к классу, который определила модель). Для NER, обученного с поддержкой пользовательского словаря, Top-N устанавливает в ответе модели количество синонимов из словаря, соответствующих найденным сущностям. Сортировка выдачи синонимов также выполняется в порядке убывания параметра **score**.



1. Введите в поле ваше сообщение и нажмите . В поле **Ответ модели** отобразится JSON с ответом. Для классификатора в нем будут label – класс, который определила модель, и score.

Для NER без словаря в json будет содержаться массив из сущностей, которые модель определила в вашем запросе. В каждом элементе массива:

text – текст запроса;

label – сущность, которую определила модель;

start/end – положение первого и последнего символа сущности, а также score.

Для NER с пользовательским словарем в ответе будет присутствовать дополнительный массив найденных синонимов **aliases** содержащий информацию из CSV-колонок словаря.

ВНИМАНИЕ

Если включен батчинг, то к ответу добавиться batchid – идентификатор батча.

2. Задайте все необходимые вопросы для проверки сервиса. Изучите ответы модели и примите решение о качестве работы сервиса.
3. Чтобы завершить тестирование, закройте виджет или выйдите из карточки сервиса.

Подключение обученной модели к проекту

Порядок подключения модели отличается в зависимости от типа сервиса.

Подключение классификатора

Чтобы подключить классификатор, обученный с помощью AutoML-сервиса:

1. На странице **Проекты** откройте конструктор нужного проекта.
2. По кнопке **Версии** перейдите в настройки текущей версии. Выберите в выпадающем списке ваш AutoML-сервис и настройте подключение. Подробнее см. в разделе «Привязка классификатора».
3. Сохраните изменения.

Подлкючение NER

Чтобы в процессе выполнения сценария бот обращался к модели NER:

1. Откройте карточку созданного сервиса типа NER и скопируйте значение поля **URL**.
2. В разделе **Проекты** откройте конструктор нужного бота.
3. Перейдите в сценарий, в котором предполагается использовать выделение сущностей из запроса клиента. Вставьте скопированный **URL** в интеграционный блок **HTTP-запрос** и заполните параметры вызова. Подробнее см. в разделе «Подключение сервиса NER».
4. Сохраните изменения.

Удаление AutoML-сервиса

Чтобы удалить AutoML-сервис:

1. Откройте карточку сервиса и нажмите **Удалить**.
2. Подтвердите удаление – начнется процесс, в ходе которого будет остановлена и удалена модель, файл датасета из хранилища S3 и записи о сервисе в базе данных.

> **ВНИМАНИЕ**
>
> Удаление сервиса необратимо.

В результате карточка сервиса закроется. Вы переместитесь на страницу **AI-сервисы**. Сервис получит статус **Удаляется**. Снова открыть его карточку не получится. Через некоторое время сервис удалится из списка сервисов.

Рекомендации по созданию датасетов для обучения классификатора

Используйте эту информацию при создании датасетов классификаторов.

Если обучаемый классификатор планируется использовать в продуктовом решении, то лучше, чтобы датасет для него содержал не менее 81 примера на класс. Кроме того, имеет значение, чтобы каждый класс в датасете был представлен примерно одинаковым количеством примеров.

Сбалансированный датасет – примерно равное количество: "positive": 500 примеров, "negative": 450 примеров, "neutral": 520 примеров. **Несбалансированный датасет** – сильный перекос: "positive": 500 примеров, "negative": 20 примеров, "neutral": 200 примеров.

(отметить) При подготовке данных обращайте внимание на их качество.

Практические советы:

1. Лучше меньше, но лучше. 20 качественных примеров дадут лучший результат, чем 200 сомнительных.
2. Проверяйте граничные случаи. Если сомневаетесь в категории – лучше исключите пример.
3. Используйте реальные данные. Примеры должны быть похожи на те тексты, которые модель увидит в работе.
4. Балансируйте специфичность. Нужны и простые типичные случаи, и более сложные варианты.

Перед добавлением примера проверьте:

- Понятно ли из текста, почему он относится к данной категории?
- Содержит ли текст конкретную информацию, а не общие слова?
- Относится ли текст только к одной категории?
- Встречаются ли подобные тексты в реальной работе?
- Правильно ли определена категория?

Придерживаетесь следующих правил заполнения csv-файла:

- Обязательные столбцы:

  text – текст для классификации на русском языке;

  label – название класса, к которому принадлежит текст. Может быть любой строкой. Название класса должно совпадать с названием сценария бота, иначе при разметке истории диалогов будет отображаться некорректно.

- Опциональный столбец:

  anc_label – якорь класса, содержит описание класса только на русском языке. Например – label: music, anc_label: музыка. Рекомендуется использовать anc_label, если хотя бы в одном из классов не более 5 примеров.

- Общие ограничения:
  - Минимальное количество классов – 2;
  - Минимальное количество примеров в классе – 2;
  - Разделитель – запятая;
  - Если в самом примере текста есть запятая, то весь пример нужно заключить в кавычки (" ").

  Пример: "можно ли построить семью с человеком, если он козерог сварщик и время рождения 14:03";

  - Если в примере есть кавычки(" "), их нужно экранировать, поставив рядом с ними дополнительные кавычки. Весь пример также нужно заключить в кавычки.

  Пример: """вкусно и точка"" доставляют сюда".

Образец правильного оформления csv-файла датасета с заполненным столбцом anc_label:

```
text,label,anc_label
закажи такси домой,taxi,такси
отвези меня к маме,taxi,такси
вызови машину на подсосненский 25/1,taxi,такси
хочу есть,food,еда
закажи пиццу,food,еда
"""вкусно и точка"" доставляют сюда",food,еда
когда меркурий будет ретроградным,astrology,астрология
рассчитай мою натальную карту,astrology,астрология
```

```
тельцы с водолеями совместимы,astrology,астрология
"можно ли построить семью с человеком, если он козерог сварщик и виталий время
рождения 14:03",astrology,астрология
```

Платформа поддерживает **OOD (Out-of-Domain)** – функцию, позволяющую модели понимать, что она столкнулась с запросом, который выходит за пределы ее обучения или специализации (то есть лежит «вне домена»). Вы можете самостоятельно определить, какие запросы будут считаться «вне домена» для обучаемой модели. При подготовке обучающего датасета таким данным необходимо присваивать метку outOfScope.

Для создания сбалансированного датасета, включающего **outOfScope**, следуйте правилам:

1. OOD не работает на низкоресурсных датасетах, когда самый минимально представленный класс содержит всего 2-5 примеров, поэтому – чем больше примеров на минимальный класс, тем лучше.
2. Если вы все же используете низкоресурсный датасет из пункта 1, требуется добавлять в примеры классов anc_label, при этом **не добавляйте примеры** outOFScope. В этом режиме система с ними не работает.
3. Если вы хотите получить лучшее качество определения примеров категории outOfScope, необходимо представить в ней примеры из следующих подкатегорий:
   - CloseOutOfScope – данные, которые близки к существующим классам по теме, но отличаются семантически.
   - MidOutOfScope– данные, которые относятся к той же общей области, что и известные модели классы, но имеют другую тематику (запросы к скиллам, которые не существуют в классификаторе)
   - FarOutOfScope – данные, которые совсем не относятся к области знаний модели.

Пример правильного оформления датасета с категорией outOfScope:

```
text;label
ответь в письме ясону что я не приду сегодня вечером;email_sendemail
alexa начни новое письмо;email_sendemail
olly отправь и-мейл мои встречи надо перенести;email_sendemail
ответить на электронное письмо роберта сегодня утром;email_sendemail
пожалуйста отправь на новый электронный адрес из списка;email_sendemail
пошли письмо моему брату и напомни годовщина свадьбы;email_sendemail
я бы хотел отправить ответ;email_sendemail
открой пожалуйста ответить на это письмо;email_sendemail
выключи свет в гараже;iot_hue_lightoff
выключи свет на кухне;iot_hue_lightoff
выключи свет;iot_hue_lightoff
выключи свет в детской спальне пожалуйста и затем измени свет в моей комнате на
красный;iot_hue_lightoff
выключи свет;iot_hue_lightoff
выключи верхний свет;iot_hue_lightoff
выключи свет в ванной;iot_hue_lightoff
выключи свет;iot_hue_lightoff
какие текущие списки у меня есть;lists_query
мой праздничный список;lists_query
как называются созданные мной списки;lists_query
что в моём списке продуктов;lists_query
названия всех списков которые я веду;lists_query
убедитесь что хлеб есть в моем списке продуктов;lists_query
я хочу чтобы вы удалили пункт из списка;lists_remove
это делает меня удачливым парнем внутри очень очень;lists_remove
```

```
удали пункт списка;lists_remove
удалить фрукты из списка;lists_remove
сотри список домашних дел;lists_remove
удалить список магазинов;lists_remove
удалить песню дельтаплан из моей музыки;lists_remove
ты знаешь этот текст;music_query
какой список доступен для моей любимой музыки мити фомина;music_query
где я могу найти эту песню;music_query
показать мне музыку того артиста;music_query
что это мы слушаем;music_query
название группы;music_query
повтори эту песню когда она закончится;outOfScope #(пример из подкатегории
closeOutOfScope)
пожалуйста поставь этот плейлист на перемешку;outOfScope #(пример из подкатегории
closeOutOfScope)
закажи китайскую кухню из тан жен и говядину с брокколи;outOfScope #(пример из
подкатегории midOutOfScope)
как выглядит моё расписание сегодня;outOfScope #(пример из подкатегории
midOutOfScope)
* боли в суставах механического типа - в пр плечевом суставе, к/с справа * боли в
суставах воспалительного типа - в пр плечевом суставе, к/с справа * боли в
позвоночнике - в поясничном отделе - облегчение в покое, усиливаются при ходьбе,
работе внаклон, периодически онемение правого бедра * ВАШ 75 см * скованность по
утрам - 15 минут в суставах * «стартовые» боли - периодически в пр к/с, ПОП *
ограничение движения в суставах - отведение в пр плечевом суставе, заведение правого
плеча за спину *  в суставах - нет;outOfScope #(пример из подкатегории farOutOfScope)
на  до 180, головную боль , головокружение;outOfScope #(пример из подкатегории
farOutOfScope)
```

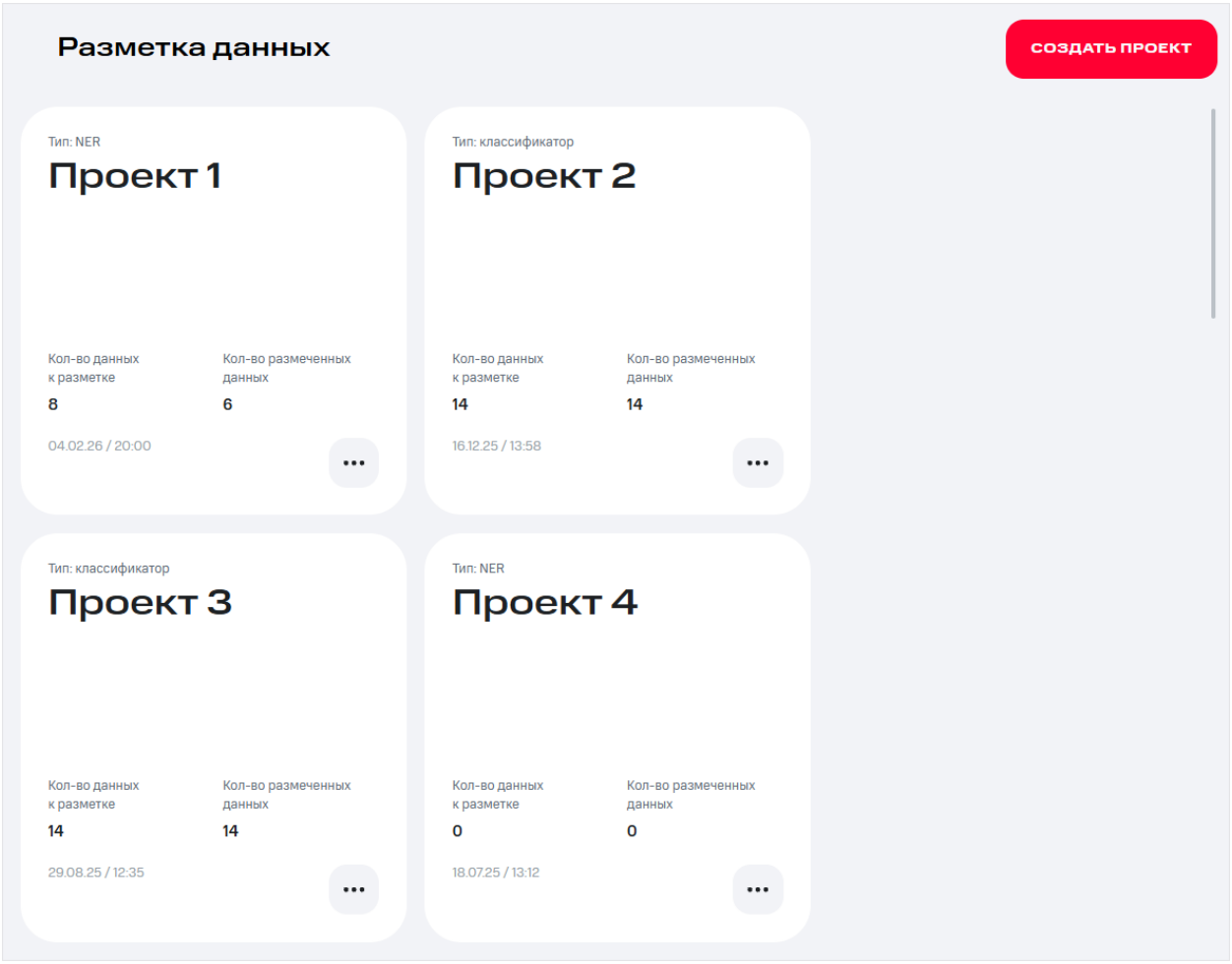
(пусто)

Разметка данных

В разделе Разметка данных вы можете создавать обучающие датасеты для ML-моделей AutoML-сервисов.

> **ВНИМАНИЕ**
>
> В процессе создания датасета для обучения модели классификатора каждому текстовому фрагменту присваивается метка, определяющая класс, к которому относится фрагмент текста. Например. фразам "запиши меня к терапевту" и "как попасть к терапевту" можно присвоить класс – therapist.
>
> Разметка данных для обучения модели NER – это выделение сущностей во фразе и присвоение каждой выделенной сущности собственной метки. Так во фразе "записаться к терапевту в Москве", можно определить сущность "в Москве" и назначить ей метку "city", а за сущностью "терапевт" закрепить метку therapist.

На главной странице раздела отображаются плитки проектов разметки и кнопка **Создать проект**.



Каждая плитка проекта содержит краткую информацию:

- тип модели, которую можно обучить на размеченных данных;
- название проекта;
- количество данных к разметке и количество уже размеченных данных;
- дату и время последнего изменения проекта.

В контекстном меню проекта доступны действия:

- редактировать проект;
- дублировать проект;
- скачать датасет проекта;
- удалить проект.

Чтобы открыть какой-либо проект для разметки или настройки, нажмите на его плитку – отобразится вкладка **Данные разметки** карточки проекта.

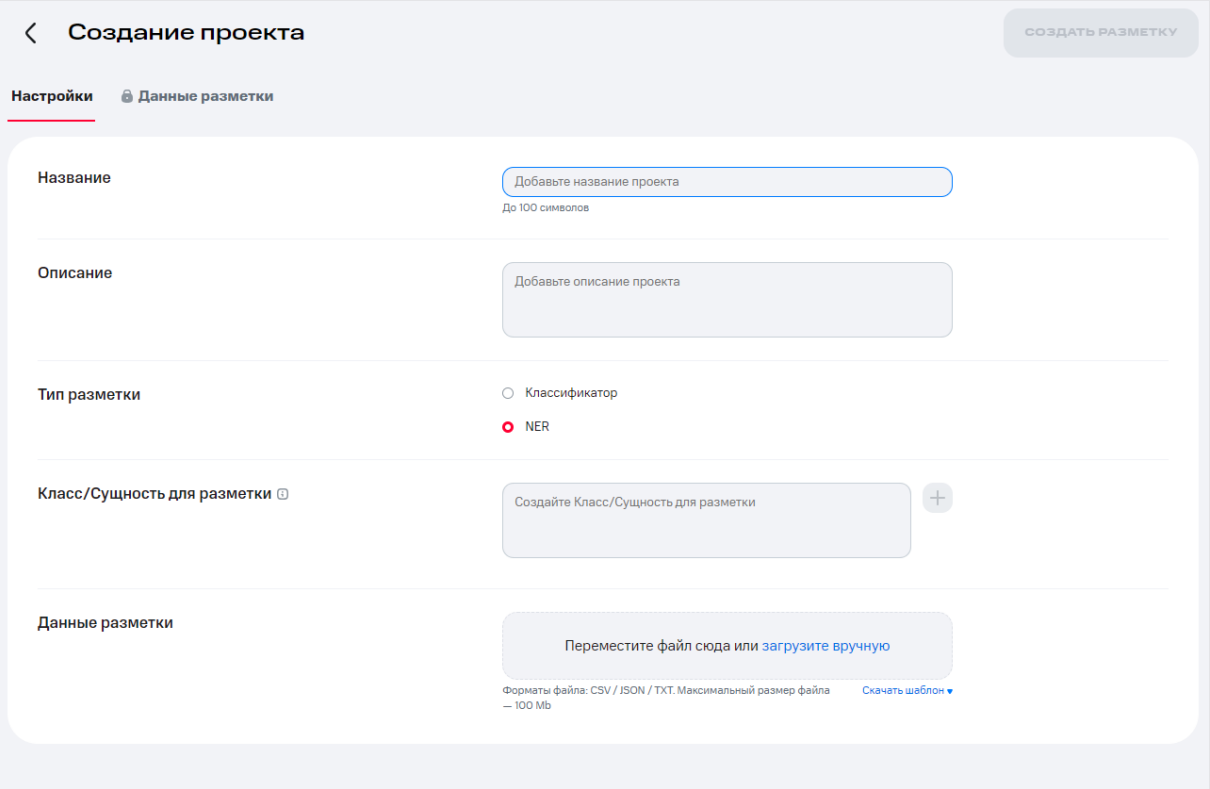
Чтобы создать обучающий датасет:

1. В разделе **Разметка данных** создайте новый проект разметки.
2. Добавьте данные для разметки.
3. Разметьте данные.
4. Скачайте датасет и используйте его в обучении AI-сервисов.

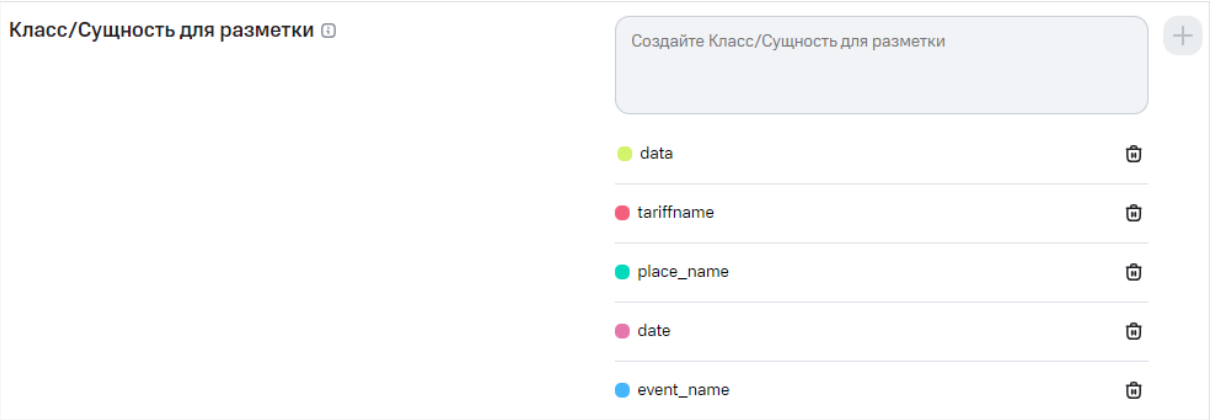
Создание проекта разметки

Чтобы создать проект разметки данных:

1. На странице **Разметка данных** нажмите кнопку **Создать проект** – откроется окно **Создание проекта**, вкладка **Настройка**. Вкладка **Данные разметки** на этапе создания заблокирована, т.к. данных еще нет.



2. Введите название проекта до 100 символов. Поле обязательно для заполнения.
3. При необходимости добавьте описание.
4. Выберите нужный тип разметки.
5. Подготовьте набор меток (лейблов) классов/сущностей, которые будут присваиваться фрагментам текста в процессе разметки данных. Для этого в поле **Класс/Сущность** для разметки через запятую (или с новой строки) введите названия меток. Добавляйте столько меток, сколько потребуется для работы с данными, количество не ограничено.
6. Нажмите кнопку **+**. Список меток появится на странице. Чтобы отредактировать метку, кликните на ней и исправьте текст. При необходимости метку можно удалить.



> **ПРИМЕЧАНИЕ**
>
> Вы можете добавлять или удалять метки классов/сущностей в процессе разметки данных.

7. Вы можете добавить в проект заранее подготовленные по шаблонам данные разметки. Для этого перетащите файл разметки в поле **Данные разметки** или загрузите его через проводник операционной системы. Форматы файла: CSV / JSON / TXT. Максимальный размер файла – 100 Мб.Чтобы получит шаблоны, нажмите **Скачать шаблон**, и выберите нужный из выпадающего списка. Вы сможете добавить, отредактировать или удалить данные в процессе разметки данных.
8. Нажмите кнопку **Создать разметку** – откроется вкладка **Данные разметки**. Вы можете начать добавление данных сразу или вернуться к проекту позже – если выйти в основной раздел **Разметка данных**, вы увидите проект в списке.

В результате созданный проект разметки отображается в списке.

Добавление данных в проект

После создания проекта разметки вы можете выбрать способ добавления данных в проект: вручную либо загрузкой файла с датасетом.

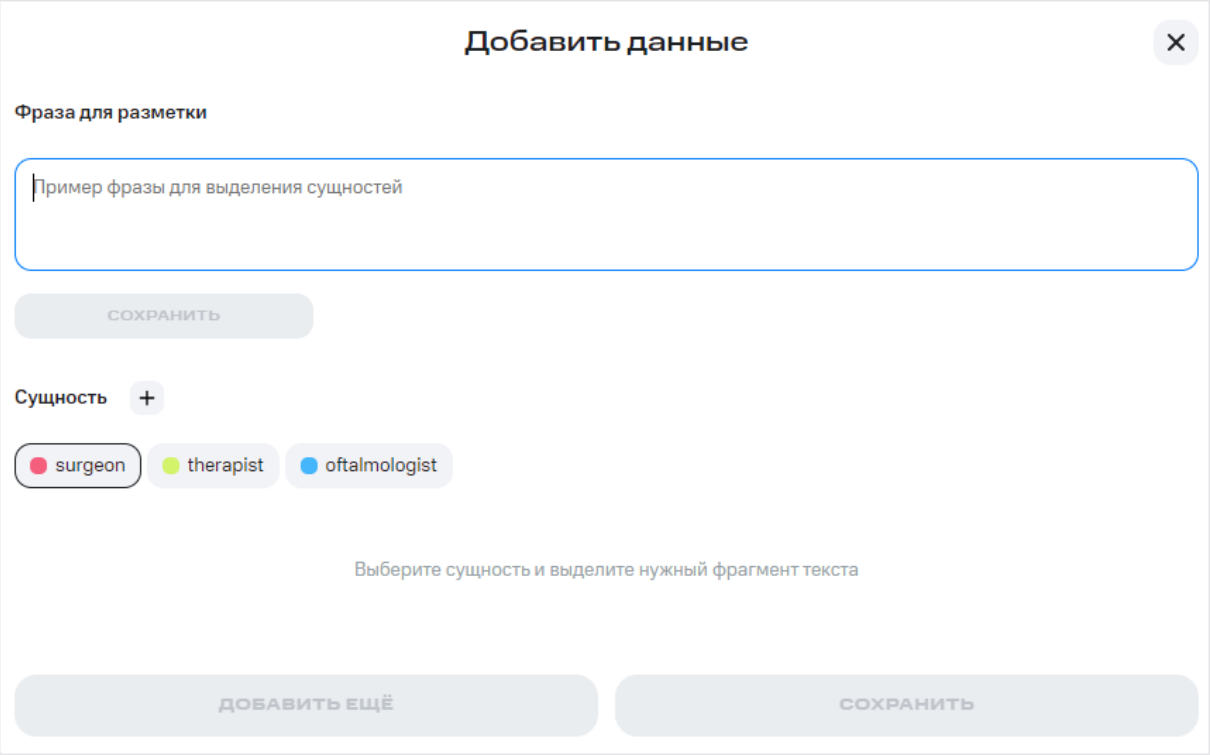
Оба способа имеют свои преимущества:

- ручной способ позволяет редактировать и размечать загружаемые фразы в процессе добавления;
- с помощью файла датасета можно загрузить значительные объемы данных.

Добавление данных вручную

Чтобы добавить данные:

1. Во вкладке **Данные проекта** нажмите кнопку **Добавить данные** – откроется одноименная форма.

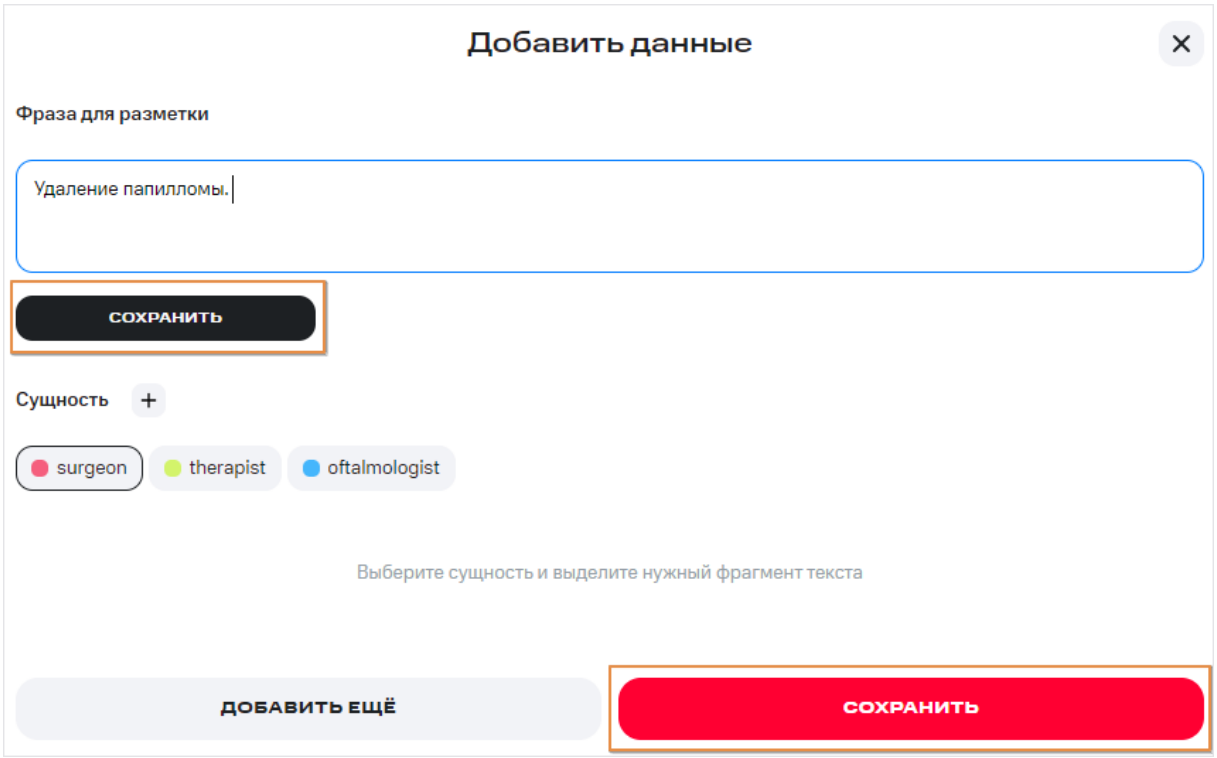


> **ПРИМЕЧАНИЕ**
>
> Если на этапе создания вы добавили в проект метки сущностей/классов, то в форме будет отображен их список. Если нет – форма будет пустой.

2. Чтобы добавить метку, нажмите на кнопку **+**. Введите в появившемся поле название и снова нажмите кнопку **+**. Новая метка появится в списке. Количество меток не ограничено.

> **ВНИМАНИЕ**
>
> Система не допускает создания двух одинаковых меток и сообщит вам, если название метки уже используется. Созданные метки можно редактировать и удалять на вкладке **Настройки**.

3. Введите фразу в соответствующее поле – станут активны черная и красная кнопки **Сохранить**:
   - Чтобы начать разметку непосредственно в текущей форме – нажмите черную кнопку **Сохранить**, разметьте фразу, как описано на следующем шаге, затем нажмите кнопку **Добавить еще**. Форма очистится для ввода новой фразы. Размеченная фраза появится в списке на вкладке **Данные разметки**.
   - Чтобы фраза сразу попала в список вкладки **Данные разметки**, нажмите красную кнопку **Сохранить** – форма закроется, фраза появится в списке.



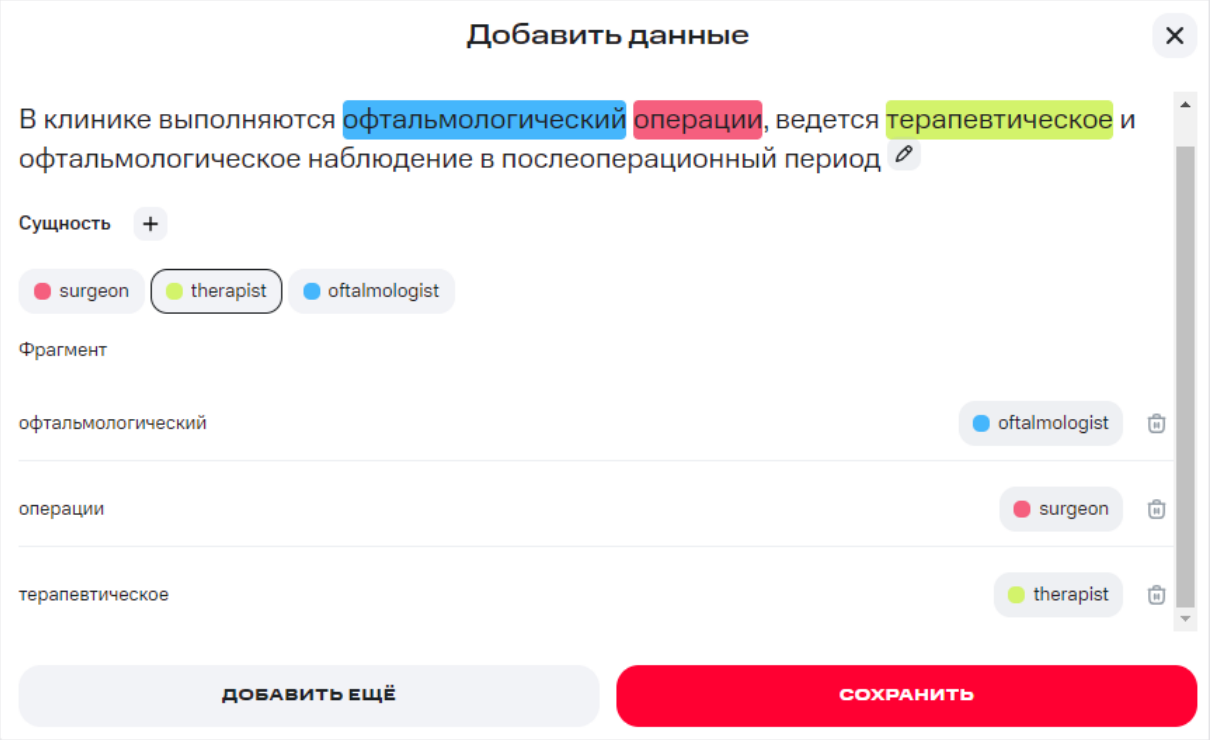
ВНИМАНИЕ

При добавлении фразы система проверяет ее уникальность. Если фраза уже существует в списке **Данные разметки**, она будет перезаписана, чтобы не произошло дублирование. Вы можете добавить неограниченное количество фраз. Чтобы отредактировать сохраненную фразу, наведите на нее курсор – появится значок . Нажмите на него, чтобы внести изменения.

4. Приступите к разметке данных в окне **Добавить данные** в процессе добавления фраз в проект.

   Разметка фразы проекта NER

   Выберите в списке метку, а затем выделите мышкой сущность во фразе, которой следует эту метку присвоить. Фрагмент подсветится цветом, идентичным цвету метки. Размеченные фрагменты будут отображаться в виде списка. Вы можете добавлять и присваивать неограниченное количество меток. При необходимости можно удалить размеченный фрагмент фразы.



   Разметка фразы проекта Классификатор

   Класс присваивается всей фразе. Поэтому вам достаточно кликнуть по нужной метке. Метка будет присвоена фразе целиком.

5. Чтобы закончить разметку, нажмите кнопку **Сохранить** – форма закроется. Размеченная фраза появится в списке на вкладке **Данные разметки**.

   сли вы хотите продолжить добавлять и размечать фразы в текущей экранной форме, нажмите кнопку **Добавить еще**. Форма очистится для ввода новой фразы. Готовая фраза отобразится в списке на вкладке **Данные разметки**.

   Количество фраз, которые можно создать и разметить в окне **Добавить данные**, не ограничено.

ПРИМЕЧАНИЕ

При добавлении размеченной фразы система проверяет уникальность ее разметки. Если фраза уже существует в списке **Данные разметки**, она будет перезаписана, чтобы не произошло дублирование. Если вы присвоили фразе-дубликату новые лейблы, они перезапишут старые. В системе отобразится соответствующее сообщение.

Загрузка файла с данными разметки

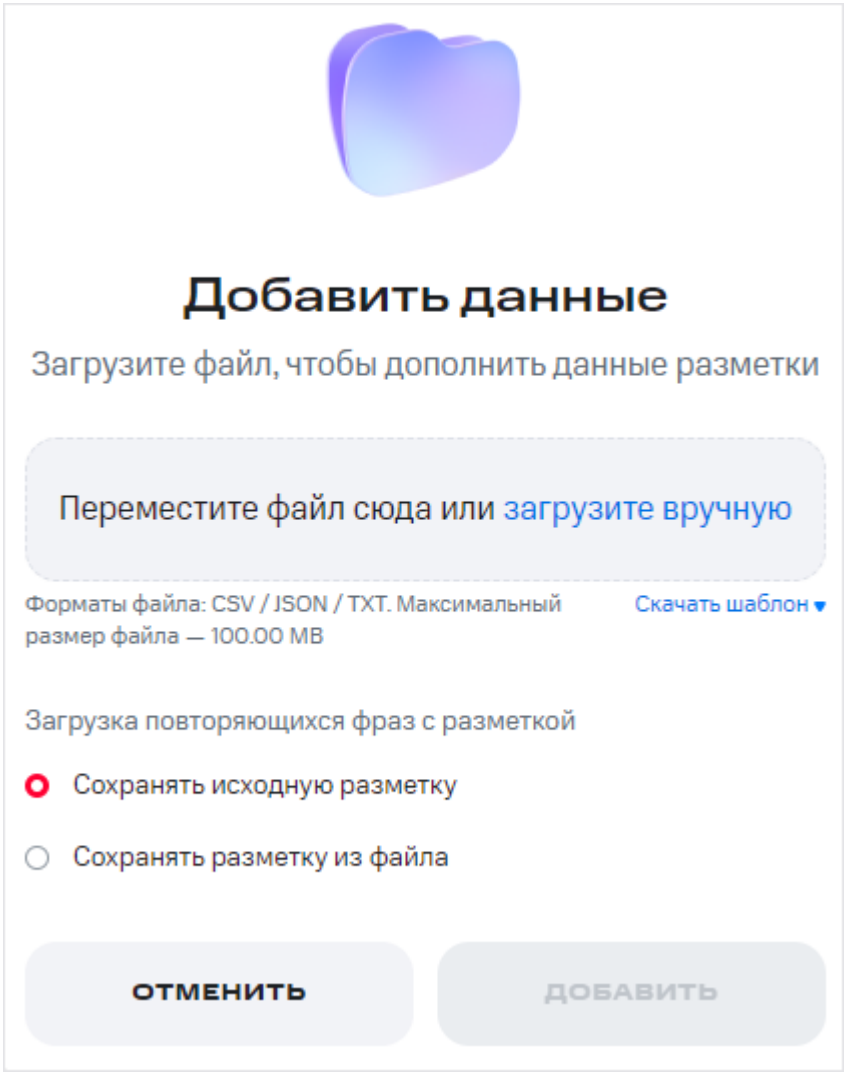
1. Нажмите кнопку **Загрузить файл** – откроется форма загрузки.

ПОДСКАЗКА

Файлы должны отвечать следующим требованиям:

- размер не более 100 MB;
- формат файла для классификатора – CSV;
- формат файла для NER – JSON;
- формат для загрузки неразмеченного текста в проект любого типа – TXT.

Написать файл разметки можно по шаблону. Чтобы получить шаблон, нажмите Скачать шаблон и выберите нужный вам тип из выпадающего списка.



2. Загрузите подготовленный файл с данными разметки: перетащите его в поле загрузки или воспользуйтесь стандартным проводником.

> **ВНИМАНИЕ**
>
> Если вы ошибочно загрузили CSV для NER или JSON для классификатора – то в проект загрузятся только фразы, разметка будет проигнорирована системой.

В окне загрузки есть опции **Сохранять исходную разметку**/**Сохранять разметку из файла**. Если файл загружается в проект, в котором уже есть данные, эти опции позволяют сохранить существующую разметку проекта или заменить ее разметкой из файла.

В процессе загрузки новых данных в дополнение к существующим, система отслеживает повторяющиеся фразы и лейблы. Сравнивается список фраз и меток, загружаемых из файла с теми, что уже есть в системе.

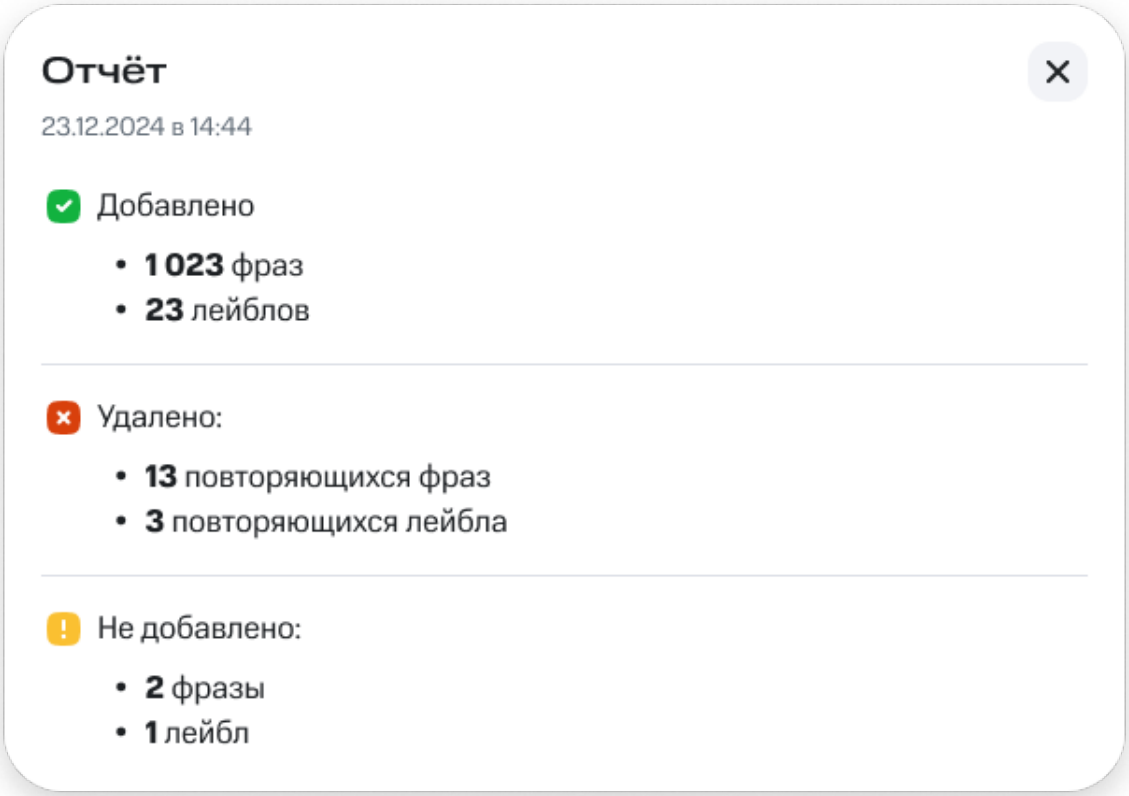
Варианты результатов сравнения:

- Если у фразы с меткой в системе обнаружен дубль с меткой, то фраза не загружается. Метка загружается без привязки к фразе, но при условии, что ее еще нет в системе.
- Если у фразы с меткой в системе обнаружен дубль без метки, то фраза не загружается. Дублю в системе присваивается метка фразы из файла данных. Метка может быть загружена в систему при условии, что ее еще нет в системе.

- Неразмеченные фразы загружаются, если у них нет дублей в системе. -Если в файле данных обнаружена метка без фразы, она будет загружена, если у нее в системе нет дубля.

Если вы загружаете файл в пустой проект, выбор опции не имеет значения – загрузятся все без исключения данные и лейблы из файла.

3. Проверьте статистику загрузки. Для этого нажмите **Посмотреть отчет**.



Разметка данных в проекте

Чтобы разметить данные в проекте:

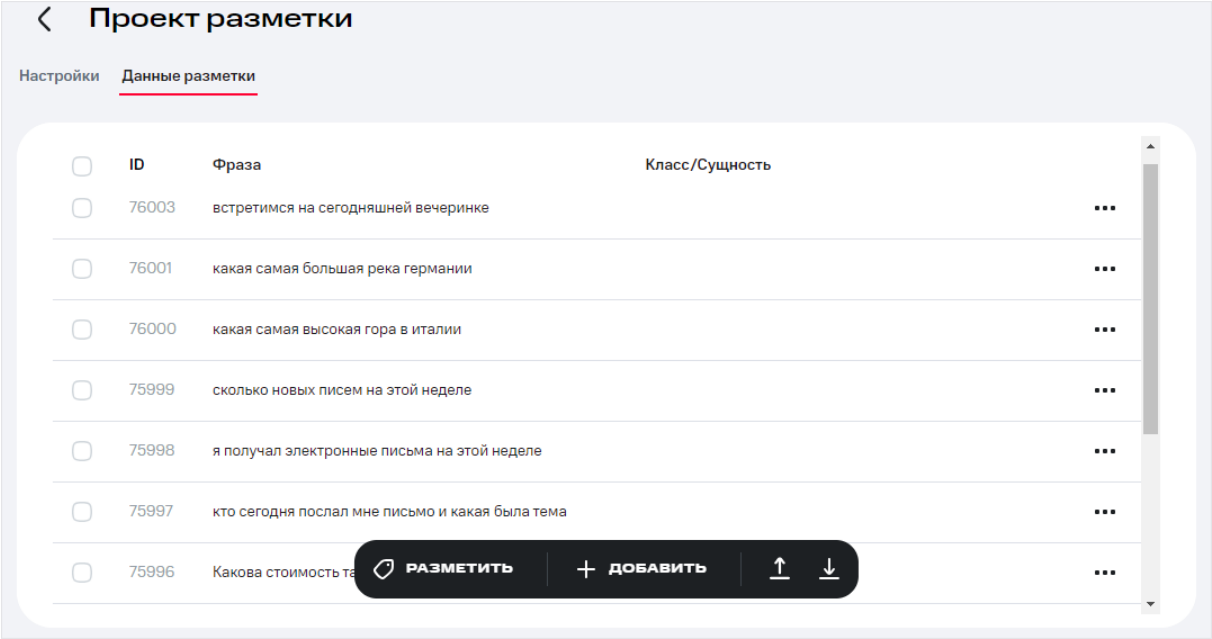
1. Откройте проект и перейдите на вкладку **Данные разметки**.

   Отображается список, в котором:

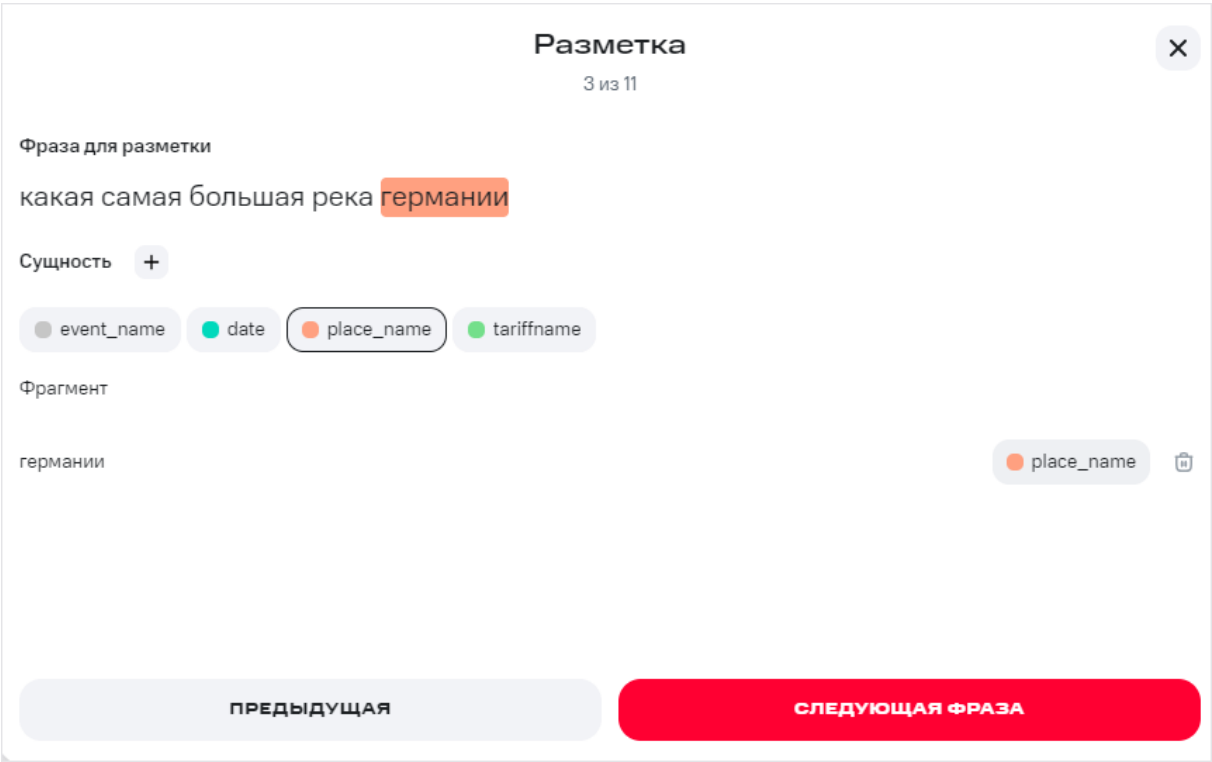
   - **id** – уникальный номер фразы в рамках проекта разметки;
   - **Фраза** – размечаемая фраза;
   - **Класс/сущность** – метки присвоенные фразе или ее сэмплам.

   > **ПРИМЕЧАНИЕ**
   >
   > Если фразы не размечены, столбец Класс/сущность будет пустым.

   Управлять списком в процессе разметки можно, вызвав контекстное меню или с помощью кнопок тулбара внизу страницы.



2. Чтобы разметить любую фразу из списка, вызовите ее контекстное меню и в нем выберите пункт **Разметить**. Либо щелкните кнопку разметить на тулбаре: в этом случае разметка начнется с первой фразы в списке. Откроется окно **Разметка**.



3. Разметьте фразу в соответствии с заданием на разметку.

   Если размечается проект NER – выберите в списке метку, а затем выделите мышкой фрагмент фразы, которому эту метку следует присвоить. Фрагмент подсветится цветом, идентичным цвету метки. Размеченные фрагменты будут отображаться внутри формы в виде списка. Вы можете добавлять и присваивать неограниченное количество меток. При необходимости можно удалить размеченный фрагмент фразы.

   Если размечаете Классификатор, выберите метку класса, который необходимо присвоить фразе.

> **ПОДСКАЗКА**
>
> Если в проекте еще нет меток, или возникла необходимость создать новые, нажмите **+** и введите в появившемся поле наименование метки, затем снова нажмите кнопку **+**. Новая метка появится в списке. Количество меток не ограничено.
>
> Чтобы отредактировать фразу, наведите на нее курсор – появится значок  . Нажмите его и приступайте к правке.

4. Чтобы перейти к следующей или предыдущей фразе, нажмите соответствующую кнопку внизу формы.
5. Повторите шаги 3 и 4 для всех фраз в списке.

Для проектов **Классификатор** в столбце **Класс/сущность** отображаются присвоенные фразам метки.

Для проектов **NER**, помимо заполненного столбца **Класс/сущность**, размеченные фрагменты фраз обозначаются подчеркиванием в цветах присвоенных меток.

Управление проектами разметки

В процессе работы с проектами разметки вы можете редактировать, добавлять или удалять как сами проекты, так и содержащиеся в них данные разметки.

На вкладке **Настройки** вы можете:

- Изменить название проекта;
- Добавить или отредактировать описание проекта;
- Добавить или удалить новые метки классов/сущностей.

Добавление данных в проект

> **ПРИМЕЧАНИЕ**
>
> Добавить данные в проект разметки можно на любом этапе работы с проектом.

Чтобы добавить в существующий проект новые данные:

1. Откройте проект и перейдите на вкладку **Данные разметки**.
2. В тулбаре нажмите кнопку **Добавить данные** – откроется одноименное окно.
3. Добавьте данные.

Загрузка данных разметки из файла

Вы можете дополнить существующий список данных новым данными из файла разметки.

Чтобы загрузить данные:

1. Откройте проект и перейдите на вкладку **Данные разметки**.
2. В тулбаре нажмите кнопку  – откроется окно **Загрузка данных**.
3. Добавьте данные.

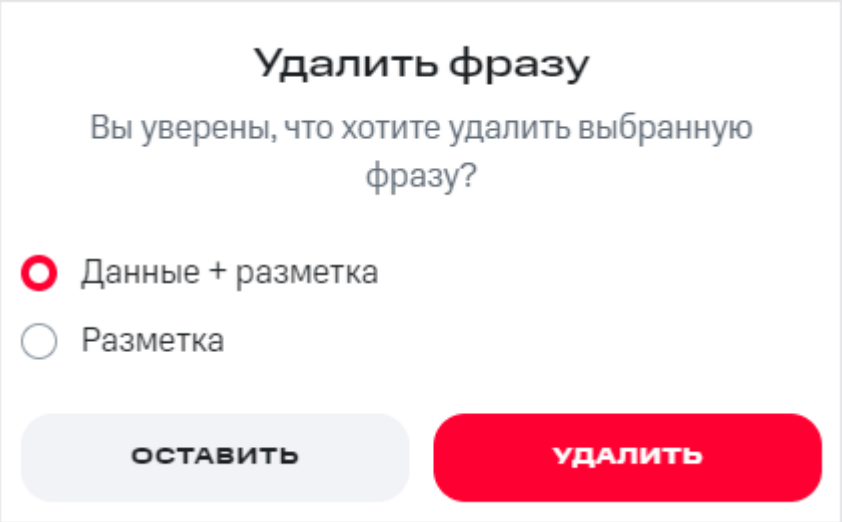
ПРИМЕЧАНИЕ

При импорте данных система проверяет наличие дубликатов фраз в проекте и файле данных. Дубли не загружаются.

В окне загрузки можно выбрать, сохранять для дублирующихся фраз текущую разметку проекта или заменить её на разметку из файла. Для этого используйте опцию **Сохранять исходную разметку** или **Сохранять разметку из файла** соответственно.

Удаление данных разметки

1. Откройте проект и перейдите на вкладку **Данные разметки**.
2. Чтобы удалить одну фразу, вызовите контекстное меню и нажмите кнопку **Удалить**. Для массового удаления выберите несколько фраз в списке или сразу весь список. Откроется диалоговое окно **Удалить фразу**:



3. Выберите способ удаления данных:
   - **Данные+разметка** – удаляются фразы вместе с разметкой, т.е. все данные проекта. Список Данные разметки станет пустым.
   - **Разметка** – удаляется привязка меток к фразам, сами фразы и метки останутся в проекте раздельно.
4. Нажмите **Удалить**.

В результате в зависимости от выбранного способа будут удалены фразы или только их разметка.

Дублирование проекта

Быстрый способ создать проект разметки – продублировать существующий.

Для этого:

1. Вызовите контекстное меню проекта в списке проектов и нажмите **Дублировать**.
2. В диалоговом окне выберите способ дублирования:
   - **Данные** – проект будет скопирован без разметки, только фразы и список меток – вы сможете переразметить данные.
   - **Данные + разметка** – в этом случае проект будет дублирован полностью.
3. Пустой проект дублируется по умолчанию, без диалогового окна. Если в нем были созданы метки – они также будут скопированы.

Проект продублирован. Новый проект появился в списке под тем же название с префиксом **Копия N**, где N – порядковый номер проекта в системе.

Скачивание датасета

Чтобы скачать результат разметки данных – датасет:

1. Откройте проект и перейдите на вкладку **Данные разметки**.
2. Отметьте фразу или выборку фраз, которые должны быть включены в датасет. Затем нажмите  . Чтобы скачать полный датасет, выберите все фразы или не выбирайте ни одной и нажмите кнопку  .

Файл с датасетом соответствующего типа скачан в папку **Загрузки**.

Удаление проекта разметки

Чтобы удалить проект разметки, вызовите его контекстное меню и нажмите **Удалить**. Подтвердите удаление.

> **ВНИМАНИЕ**
>
> Все данные разметки, связанные с проектом, будут удалены.