

УТВЕРЖДЕН

DOC-XX.XXX.XXX.XXX -ЛУ

Этот документ содержит спецификацию API подсистемы «Диалоговый ассистент» — интеллектуального веб-интерфейса, разработанного для эффективной поддержки пользователей на базе государственных ИТ-систем. В документе представлены методы, параметры, форматы запросов и ответов, обеспечивающие интеграцию внешних систем с ассистентом. Спецификация создана для разработчиков и интеграторов, чтобы упростить подключение, обеспечить корректную работу и масштабировать решения. Система повышает доступность цифровых сервисов, снижает нагрузку на операторов и улучшает пользовательский опыт.

ПОДСИСТЕМА «ОНЛАЙН-КОНСУЛЬТАНТ»

ФЕДЕРАЛЬНОЙ ГОСУДАРСТВЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Описание программного интерфейса «Service API»

DOC-XX.XXX.XXX.XXX

Государственный контракт от 32 марта 20XX г. № XX.XXXXX.XXX

Листов 31

Москва-20XX
<https://example.org/>

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

Настоящий документ содержит Описание программного интерфейса «Service API¹» Подсистемы «Онлайн-консультант» Федеральной Государственной Информационной Системы. (далее по тексту ОК²).

Документ оформлен в соответствии с правилами, предусмотренными ГОСТ Р 59795–2021, ГОСТ 2.301-68, ГОСТ 34.201-2020, ГОСТ 19.105, ГОСТ 19.106 и другими стандартами Единой системы программной документации (ЕСПД).

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
Изм.	Лист	№ докум.	Подп.	Дата

¹ Service API (SAPI) v1.0
² ОК – Онлайн консультант

СОДЕРЖАНИЕ

АННОТАЦИЯ.....	2
СОДЕРЖАНИЕ.....	3
1 Введение.....	4
2 Требования подключения	5
2.1 Базовые понятия и определения	5
3 Авторизация	6
3.1 Инициализация клиента.....	6
3.2 Ведение диалога с собеседником	7
4 Общие структуры данных	9
4.1.1 Контекст (context).....	9
5 Методы Сервиса.....	12
5.1 Общие параметры API	12
5.1.1 Протокол.....	12
5.1.2 Headers	12
5.2 Установка WebHook'a (/setWebhook)	12
5.3 Проверка WebHook'a (/getWebhook)	14
5.4 Отправка сообщения (/sendRequest)	16
5.5 Отправка события (/sendEvent).....	18
5.6 Отправка события (/event). Deprecated	19
5.7 Оценка ответа ассистента (/rateResponse)	20
5.8 Ответ Business Logic Service (/bls/response).....	22
6 Служебные методы	27
6.1 Проверка состояния сервиса (/health_check)	27
6.2 Запуск/перезапуск сервиса (/startup)	27
6.3 Проверка готовности сервиса к обработке запросов(/readiness)	28
6.4 Проверка активности сервиса и его компонентов (/liveness).....	28
6.5 Перенаправление к документации (/)	29
6.6 Предоставление метрик (/metrics)	29
6.7 Поддерживаемые события (Events List)	30

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата						Лист 3	
Изм.	Лист	№ докум.	Подп.	Дата							

5.3	Проверка Webhook а (/getWebhook).....	14
5.4	Отправка сообщения (/sendRequest).....	16
5.5	Отправка события (/sendEvent).....	18
5.6	Отправка события (/event). Deprecated	19
5.7	Оценка ответа ассистента (/rateResponse)	20
5.8	Ответ Business Logic Service (/bls/response).....	22
6	Служебные методы	27
6.1	Проверка состояния сервиса (/health_check)	27
6.2	Запуск/перезапуск сервиса (/startup)	27
6.3	Проверка готовности сервиса к обработке запросов(/readiness)	28
6.4	Проверка активности сервиса и его компонентов (/liveness).....	28
6.5	Перенаправление к документации (/).....	29
6.6	Предоставление метрик (/metrics)	29
6.7	Поддерживаемые события (Events List)	30

					DOC-XX.XXX.XXX.XXX	Лист 3

1 ВВЕДЕНИЕ

В настоящем документе содержится описание программного интерфейса «ОК Service API» (далее – SAPI), предназначенного для подключения в качестве шлюза обмена текстовыми сообщениями к сервису диалоговых ассистентов ОК.

SAPI представляет собой асинхронный программный интерфейс подключения клиентского приложения или сервиса (далее - клиент) к ОК для обмена текстовыми сообщениями с ассистентом.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата						
Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX					Лист
										4

2 ТРЕБОВАНИЯ ПОДКЛЮЧЕНИЯ

Для использования интерфейса SAPI необходимо обладать следующими данными:

- DSN интерфейса SAPI (для публичного облака: <https://sapi.dos.nanosemantics.ai>), далее – SAPI-DSN
- Данные авторизации
- Идентификатор канала (ChannelUID) ассистента в ОК к которому производится подключение (его можно получить в интерфейсе ОК путем создания канала с типом Server API). Данный канал должен быть активирован.

2.1 Базовые понятия и определения

Ниже приведен краткий список определений, используемых при работе с ОК и SAPI.

- Собеседник – конечный пользователь системы, ведущий диалог с ассистентом.
- Диалог – обмен сообщениями между конкретным Собеседником и ассистентом.

Диалог разбивается на последовательные Сессии разделенные периодом отсутствия активности Собеседника.

- ChannelUID – уникальный идентификатор (UUID) канала ассистента в ОК (его можно создать через DOS Logic UI).
- DialogUID – уникальный идентификатор (UUID) диалога Собеседника с ассистентом.
- SAPI-DSN - DSN интерфейса SAPI.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
Изм.	Лист	№ докум.	Подп.	Дата
DOC-XX.XXX.XXX.XXX				Лист
				5

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Процесс авторизации в API обычно включает в себя несколько этапов:

- Этот процесс может отличаться в зависимости от конкретного API и требований к безопасности. Некоторые API могут использовать дополнительные методы проверки подлинности, такие как двухфакторная аутентификация.

Перед подключением Клиента к ОК через интерфейс SAPI (Рисунок 1) необходимо установить webhook на который по протоколу HTTPs будут отправляться все сообщения от ОК. Установка webhook'a производится с помощью вызова функции /setWebhook, в результате которого для указанного канала целевого ассистента ОК будет зарегистрирован указанный DSN для передачи сообщений от ОК.

Для проверки ранее установленных параметров webhook можно использовать запрос `/getWebhook`.

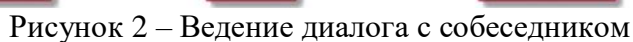


3.2 Введение диалога с собеседником

После инициализации диалога имеет смысл запросить приветственное сообщение. Приветственное сообщение возвращается в ответ на событие начала диалога.

- 1) Передача текстового сообщения от Собеседника: Выполняется через вызов метода `/sendRequest`, который позволяет отправить текстовое сообщение ассистенту в рамках текущего диалога.
- 2) Передача события диалога: Выполняется через вызов метода `/sendEvent`, который позволяет отправить событие ассистенту в рамках текущего диалога.
- 3) Оценка ранее полученного ответа ассистента: Выполняется через вызов метода `/rateResponse`, который позволяет оценить ранее полученный ответ от ассистента в рамках текущего диалога.
- 4) Получение ответа от ассистента: Сообщение от ассистента отправляется на `webhook`, установленный через вызов метода `/setWebhook` или переданный в качестве

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата



Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Передача файлов (в настоящее время в виде url)

- Данные DSN для отправки сообщений от ассистента (reply_to)

- ### 4.1.1 Контекст (context)

Переменные контекста используются для передачи информации, необходимой для формирования ответов на запросы к ассистенту и поддержания общего диалога.

Сообщение ассистента (response)

Сообщение ассистента представляет собой массив элементов, описанных ниже:

а) текст

Элемент предназначен для отображения текстовых данных.

Элемент представляет собой JSON структуру со следующими полями:

- type – {type: string}: text – фиксированное значение типа элемента.
- text – {type: string} – текстовые данные для отображения.

б) кликабельный ответ

Элемент предназначен для отображения кликабельного текстового элемента, клик по которому приводит к автоматической отправке сообщения ассистенту.

Элемент представляет собой JSON структуру со следующими полями:

- type – {type: string}: userlink – фиксированное значение типа элемента.
- ink – {type: string} – текстовые данные для отображения.
- request – {type: string, optional: true} – запрос, отправляемый ассистенту при клике по элементу. Если это поле отсутствует или не заполнено, то вместо него используется значение поля link.

в) ссылка

Элемент предназначен для отображения кликабельной ссылки.

Элемент представляет собой JSON структуру со следующими полями:

- type – {type: string}: link – фиксированное значение типа элемента.
- link - {type: string} – текст для отображения. Если это поле отсутствует, то вместо него используется полный адрес ссылки.
- ref - {type: string} - адрес документа (URL, Universal Resource Locator, универсальный указатель ресурсов), на который следует перейти при клике по элементу.
- target - {type: string} - имя окна или фрейма, куда браузер будет загружать документ.

г) перенос строки

Элемент предназначен для начала новой строки в рамках ответа ассистента.

Элемент представляет собой JSON структуру со следующими полями:

- type – {type: string}: br – фиксированное значение типа элемента.

д) изображение

Элемент предназначен для отображения изображений в графическом формате gif, jpeg или png. Адрес файла с картинкой задаётся через атрибут src.

Элемент представляет собой JSON структуру со следующими полями:

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.		Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
																10

- `type – {type: string}: img` – фиксированное значение типа элемента.
- `src – {type: string}` – путь к графическому файлу.
- `alt – {type: string, optional: true}` – альтернативный текст для изображения.

е) список

Элемент предназначен для отображения набора маркированного списка.

Элемент представляет собой JSON структуру со следующими полями:

- `type – {type: string}: list` – фиксированное значение типа элемента.
 - 1) `ordered – {type: bool, optional: true}` – признак нумерованного списка. По умолчанию: `true`.
 - 2) `items – {type: array}` – массив элементов списка, где каждый элемент представляет собой JSON структуру со следующими полями:
 - 3) `type – {type: string}: item` – фиксированное значение типа элемента.
 - 4) `values - {type: array[response]}` – массив элементов списка, представленных объектами типа `response`.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата						
Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX					Лист
										11

5 МЕТОДЫ СЕРВИСА

5.1 Общие параметры API

5.1.1 Протокол

В качестве транспорта для взаимодействия с SAPI используется HTTPS с TLS шифрованием (HTTP не поддерживается). Данные передаются в формате JSON в кодировке UTF-8.

5.1.2 Headers

При всех обращениях к SAPI необходимо использовать следующие HTTP-заголовки:

- Content-Type: application/json

5.2 Установка WebHook'a (/setWebhook)

Настройка адреса для отправки сообщений передаваемых ОК.

При отправке любого запроса на webhook устанавливается заголовок X-NLab-WebHook-Key со значением key, который позволят идентифицировать источник оповещения на принимающей стороне.

В процессе обработки запроса производится HTTP запрос к указанному webhook'у, в ответ на который ожидается ответ с HTTP статус кодом 200 и телом, совпадающим со значением параметра verify. Если сервис ответит другим кодом или иным телом ответа, то проверка будет считаться неуспешной.

5.2.1.1 Запрос

а) Method

- POST: {SAPI-DSN}/api/v1/setWebhook/{ChannelUID}

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.		Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
																12

6) Arguments

- ChannelUID – {type: UUID} – идентификатор канала ассистента, для которого устанавливается webhook.

B) Request Body

Тело запроса представляет собой JSON структуру со следующими полями:

- `url - {type: string}` - адрес (DSN) для отправки сообщений от ассистента.
- `key - {type: string}` - ключ, передаваемый в заголовке X-NLab-WebHook-Key вместе с каждым сообщением.
- `verify - {type: string}` - строка, которую сервис должен возвращать в качестве body при проверке webhook's.

5.2.1.2 Результаты

a) Success

- HTTP Code: 200
 - 1) Тело ответа представляет собой JSON структуру со следующими полями:
`success – {type: bool}: true` – фиксированное значение.

6) Error

- HTTP Code: 4** / 500
- 1) Тело ответа представляет собой JSON структуру со следующими полями:
 - а) `success - {type: bool} : false` – фиксированное значение.
 - б) `result - {type: object}`:
 - в) `error_type - {type: string}` - тип ошибки.
 - г) `error_message - {type: string}` - описание ошибки.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
<p>- HTTP Code: 200</p> <p>1) Тело ответа представляет собой JSON структуру со следующими полями: success – {type: bool}: true – фиксированное значение.</p> <p>б) Error</p> <p>- HTTP Code: 4** / 500</p> <p>1) Тело ответа представляет собой JSON структуру со следующими полями:</p> <p>а) success – {type: bool}: false – фиксированное значение.</p> <p>б) result - {type: object}:</p> <p>в) error_type - {type: string} - тип ошибки.</p> <p>г) error_message - {type: string} - описание ошибки.</p>				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата
Изм.	Лист	№ докум.	Подп.	Дата
DOC-XX.XXX.XXX.XXX				
Лист 13				

5.3 Проверка WebHook'a (/getWebhook)

Проверка установленного webhook, на который по протоколу HTTPs будут отправляться все сообщения от ОК.

5.3.1.1 Запрос

а) Method

– POST: {SAPI-DSN}/api/v1/getWebhook/{ChannelUID}

б) Arguments

– ChannelUID – {type: UUID} – идентификатор канала ассистента, для которого запрашиваются данные webhook's.

5.3.1.2 Результаты

а) Успех

- HTTP Code: 200

1) Тело ответа представляет собой JSON структуру со следующими полями:

а) success – {type: bool}: true – фиксированное значение.

б) url – {type: string} – адрес (DSN) для отправки сообщений от ассистента.

в) key – {type: string} – ключ, передаваемый в заголовке X-NLab-WebHook-Key вместе с каждым сообщением.

г) verify – {type: string} – строка, которую сервис должен возвращать в качестве body при проверке webhook's.

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.		Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
																14

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

- ### a) Method

- ## 6) Arguments

- ## B) Request Body

- `context - {type: json}` - настройка начального контекста диалога.
- 1) `extra * - {type: json}` - дополнительные параметры. deprecated
- 2) `external_user_id * - {type: string}` - id внешнего пользователя чата (перемещено в **** extra ****)

Инв. № подл.	Подп. и дата
Взам. инв. №	Инв. № дубл.
Подп. и дата	
Инв. № подл.	

5.4.1.1 Запрос

а) Method

- POST: **{SAPI-DSN}/api/v1/sendRequest/{ChannelUID}/{DialogUID}**

б) Arguments

- ChannelUID – {type: UUID} – идентификатор канала ассистента, в рамках которого происходит диалог.
- DialogUID – {type: UUID} – идентификатор диалога.

в) Request Body

Тело запроса представляет собой JSON структуру со следующими полями:

- message – {type: string} – текст сообщения ассистенту.
- context – {type: context, optional: true} – контекст сообщения.
- reply_to – {type: reply_to, optional: true} – адрес (DSN) для отправки ответных сообщений от ассистента для конкретного сообщения.

- 1) files - {type: files}

5.4.1.2 Результаты

а) Успех

- HTTP Code: 200

- 1) Тело ответа представляет собой JSON структуру со следующими полями:

- а) success – {type: bool}: true – фиксированное значение.
- б) reqid – {type: UUID} – уникальный идентификатор сообщения.

б) Ошибка

- HTTP Code: 4** / 500

Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
						17

Инв. № подл.	Подп. и дата
Взам. инв. №	Инв. № дубл.
Подп. и дата	
Изм.	Лист
№ докум.	Подп.
Дата	

5.5.1.2 Результаты

а) Успех

- HTTP Code: 200
 - 1) Тело ответа представляет собой JSON структуру со следующими полями:
 - а) `success – {type: bool}: true` – фиксированное значение.
 - б) `reqid – {type: UUID}` – уникальный идентификатор события.

б) Ошибка

- HTTP Code: 4** / 500
 - 1) Тело ответа представляет собой JSON структуру со следующими полями:
 - а) `success – {type: bool}: false` – фиксированное значение.
 - б) `result - {type: object}`:
 - в) `error_type - {type: string}` - тип ошибки.
 - г) `error_message - {type: string}` - описание ошибки.

5.6 Отправка события (/event). Deprecated

Отправка события ассистенту.

Метод /event больше не рекомендуется использовать.

5.6.1.1 Запрос

а) Method

- POST: `{SAPI-DSN}/api/v1/event/{ChannelUID}/{DialogUID}`

б) Arguments

- `ChannelUID – {type: UUID}` – идентификатор канала ассистента, в рамках которого происходит диалог.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

- DialogUID – {type: UUID} – идентификатор диалога.

в) Request Body

Тело запроса представляет собой JSON структуру со следующими полями:

- event_uid – {type: UUID} – идентификатор события.
- context – {type: context, optional: true} - контекст сообщения.
- reply_to – {type: reply_to, optional: true} – адрес (DSN) для отправки

ответных сообщений от ассистента для конкретного сообщения.

- 1) files – {type: array[file]}

5.6.1.2 Результаты

а) Успех

- HTTP Code: 200
 - 1) Тело ответа представляет собой JSON структуру со следующими полями:
 - а) success – {type: bool}: true – фиксированное значение.
 - б) reqid – {type: UUID} – уникальный идентификатор события.

б) Ошибка

- HTTP Code: 4** / 500
- Тело ответа представляет собой JSON структуру со следующими полями:
- success – {type: bool}: false – фиксированное значение.
- result - {type: object}:
 - 1) error_type – {type: string} - тип ошибки.
 - 2) error_message – {type: string} – описание ошибки.

5.7 Оценка ответа ассистента (/rateResponse)

Оценка полученного от ассистента ответа.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата	DOC-XX.XXX.XXX.XXX				Лист
									20
Изм.	Лист	№ докум.	Подп.	Дата					

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

- POST: {**SAPI-DSN**}/api/v1/rateResponse/{**ChannelUID**}/{**DialogUID**}

- **ChannelUID** – {type: UUID} – идентификатор канала ассистента, в рамках которого происходит диалог.
- **DialogUID** – {type: UUID} – идентификатор диалога.

Тело запроса представляет собой JSON структуру со следующими полями:

- `reqid` – {type: UUID} – идентификатор сообщения или события.
- 1) `context` – {type: context, optional: true} – контекст сообщения.
 - а) `reply_to` – {type: reply_to, optional: true} – адрес (DSN) для отправки ответных сообщений от ассистента для конкретной оценки.
 - б) `rate` – {type: int, range: [0, 9]} – оценка ответа ассистента.
 - в) `comment` – {type: string} – комментарий к оценке.

а) Успех

- HTTP Code: 200
 - 1) Тело ответа представляет собой JSON структуру со следующими полями:
 - а) `success – {type: bool}: true` – фиксированное значение.
 - б) `reqid – {type: UUID}` – уникальный идентификатор оценки.

б) Ошибка

- HTTP Code: 4** / 500

1) Тело ответа представляет собой JSON структуру со следующими полями:

2) `success - {type: bool}: false` – фиксированное значение.

3) `result - {type: object}`:

4) `error_type - {type: string}` - тип ошибки.

5) `error_message - {type: string}` – описание ошибки.

5.8 Ответ Business Logic Service (/bls/response)

Ответ, полученный от ассистента.

/bls/response является внутренним методом, который не рекомендуется использовать во избежание негативных последствий.

5.8.1.1 Запрос

а) Method

- POST: **{SAPI-DSN}/api/v1/bls/response**

б) Headers

- `kind - {type: string}` - Тип сообщения в очереди (`kind: request` - ответ на запрос, `kind:`

- `session.close` - уведомление о закрытии сессии, `kind: session.switch.assistant` - уведомление о переходе сессии с ассистента на оператора)

- `trace_id - {type: string}` - Идентификатор запроса в Jaeger

- `client_auth_key - {type: string, optional: true}` – Клиентский ключ

1) `client_host - {type: string}` – Адрес клиента

- `reqid - {type: UUID}` – Идентификатор запроса

- `content_type - {type: string, optional: true}` – `application/json`

- `destination - {type: string, optional: true}` - Целевой адресат в RabbitMQ

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.		Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
																22

- **duid** – {type: UUID} – Идентификатор диалога
 - 1) **buid** – {type: UUID} – Идентификатор канала
 - 2) **suid** – {type: UUID} – Идентификатор сессии
- **euid** – {type: UUID} – Идентификатор события

в) Request Body

Тело запроса представляет собой JSON структуру со следующими полями:

– **context** – {type: object} - переменные контекста используются для передачи информации, необходимой для формирования ответов на запросы к ассистенту и поддержания общего диалога

– **text** – {type: object, optional: true} - элемент обозначает текстовое сообщение

1) **value** – {type: string, optional: true} - текст сообщения

– **euid** – {type: UUID, optional: true} – идентификатор события

1) **meta** – {type: object} – метаданные сообщения

– **raw_answer** – {type: array} – сообщение ассистента (response)

1) **InfItem** – {type: object} – элемент предназначен для отображения кликабельного текстового элемента, клик по которому приводит к автоматической отправке сообщения ассистенту

a) **type** – {type: string, optional: true} – userlink – фиксированное значение типа элемента

б) **link** – {type: string, optional: true} – текстовые данные для отображения

в) **request** – {type: string} – запрос, отправляемый ассистенту при клике по элементу. Если это поле отсутствует или не заполнено, то вместо него используется значение поля link

– **HrefItem** – {type: object} – элемент предназначен для отображения кликабельной ссылки

1) **type** – {type: string} – link – фиксированное значение типа элемента

2) **ref** – {type: string, optional: true} – текст для отображения. Если это поле отсутствует, то вместо него используется полный адрес ссылки

3) **link** – {type: string} – адрес документа (URL, Universal Resource Locator, универсальный указатель ресурсов), на который следует перейти при клике по элементу

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.	
Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX				Лист
									23

4) `target` – {`type`: `string`, `optional`: `true`} – имя окна или фрейма, куда браузер будет загружать документ

- `PreItem` – {`type`: `object`} – элемент используется для указания, что текст внутри элемента должен отображаться с сохранением пробелов и переносов строк так, как они заданы изначально

1) `type` – {`type`: `string`} – `pre` – фиксированное значение типа элемента

2) `text` – {`type`: `string`} – преформатированный текст

- `HtmlItem` – {`type`: `object`} – элемент предназначен для начала новой строки в рамках ответа ассистента

1) `type` – {`type`: `string`} – `br` – фиксированное значение типа элемента

a) `tag` – {`type`: `string`, `optional`: `true`} – HTML-тег

- `TextItem` – {`type`: `object`} – элемент предназначен для отображения текстовых данных

1) `type` – {`type`: `string`} – `text` – фиксированное значение типа элемента

2) `text` – {`type`: `string`} – текстовые данные для отображения

3) `parsed` – {`type`: `bool`, `optional`: `true`} – обработанный текст.

`False` – фиксированное значение

- `ImgItem` – {`type`: `object`} – элемент предназначен для отображения изображений в графическом формате gif, jpeg или png

1) `type` – {`type`: `string`} – `img` – фиксированное значение типа элемента

2) `src` – {`type`: `string`} – путь к графическому файлу

3) `alt` – {`type`: `string`} – альтернативный текст для изображения

- `IntegrationItem` – {`type`: `object`} – элемент предназначен для отображения информации об интеграции

1) `type` – {`type`: `string`} – `dis` – фиксированное значение типа элемента

2) `name` – {`type`: `string`} – имя интегрируемого сервиса

3) `method` – {`type`: `string`} – метод интегрируемого сервиса

4) `arg` – {`type`: `object`} – словарь, содержащий дополнительные аргументы, необходимые для выполнения интеграции

- `ListItem` – {`type`: `array`} – элемент предназначен для отображения набора маркированного списка

1) `type` – {`type`: `string`} – `list` – фиксированное значение типа элемента

2) `ordered` – {`type`: `bool`} – признак нумерованного списка. По умолчанию: `true`

Подп. и дата		Инв. № дубл.		Взам. инв. №		Подп. и дата		Инв. № подл.	
Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX				Лист
									24

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Подп. и дата

ИНВ. № дубл.	
--------------	--

Взам. инв. №	
--------------	--

Подп. и дата

ИНВ. № подл.	
--------------	--

Изм.	Лист	№ докум.	Подп.	Дата

DOC-XX.XXX.XXX.XXX

Лист
25

а) Успех

3) `result` - {type: string} - результат обработки запроса.

Формат А4

- б) error_message – {type: string} – описание ошибки.
- в) error_data – {type: string} – дополнительная информация об ошибке.

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
						26

6 СЛУЖЕБНЫЕ МЕТОДЫ

6.1 Проверка состояния сервиса (/health_check)

Метод выполняет базовую проверку работоспособности приложения.

6.1.1.1 Запрос

a) Method

GET: {SAPI-DSN}/health_check

6.1.1.2 Результаты

a) Успех

HTTP Code: 200

6.2 Запуск/перезапуск сервиса (/startup)

Метод позволяет выполнить необходимые действия для инициализации сервиса.

6.2.1.1 Запрос

a) Method

GET: {SAPI-DSN}/startup

6.2.1.2 Результаты

a) Успех

HTTP Code: 200

Инв. № подл.	Подп. и дата
Взам. инв. №	Инв. № дубл.
Подп. и дата	
Инв. № подл.	

Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
						27

6.3 Проверка готовности сервиса к обработке запросов(/readiness)

Метод предоставляет информацию о состоянии сервиса, включая готовность основных компонентов.

6.3.1.1 Запрос

a) Method

GET: {SAPI-DSN}/readiness

6.3.1.2 Результаты

a) Успех

HTTP Code: 200

6.4 Проверка активности сервиса и его компонентов (/liveness)

Метод используется для проверки жизнеспособности (лайвнесса) сервиса. Он помогает определить, активен ли сервис и работает ли он должным образом.

6.4.1.1 Запрос

a) Method

GET: {SAPI-DSN}/liveness

6.4.1.2 Результаты

a) Успех

HTTP Code: 200

Инв. № подл.	Подп. и дата
Взам. инв. №	Инв. № дубл.
Подп. и дата	
Инв. № подл.	

Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
						28

6.5 Перенаправление к документации (/)

Метод представляет путь к документации.

6.5.1.1 Запрос

a) Method

GET: {SAPI-DSN}/

6.5.1.2 Результаты

a) Успех

HTTP Code: 200

6.6 Предоставление метрик (/metrics)

Метод используется для сбора и предоставления метрик и статистики работы сервиса.

6.6.1.1 Запрос

a) Method

GET: {SAPI-DSN}/metrics

6.6.1.2 Результаты

a) Успех

HTTP Code: 200

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX	Лист
						29

6.7 Поддерживаемые события (Events List)

Событие начала диалога (00b2fcbe-f27f-437b-a0d5-91072d840ed3)

Формат сообщений, отправляемых на установленный webhook (Response format)

Сообщения, передаваемые через Webhook, представляют собой JSON структуру одного из описанных ниже типов.

В каждом сообщении обязательно присутствуют следующие поля:

- **type** * - {type: string}: message – фиксированное значение.
- **dialog_uid** * - {type: uuid} – уникальный идентификатор диалога. *
- **message** * - {type: response} – сообщение для передачи собеседнику.
- **context** * - {type: context} – изменение контекста диалога.
- **attachments** * - {type: attachments} - вложения
- 1) **files** * - {type: array[file]} - структура

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата					
Изм.	Лист	№ докум.	Подп.	Дата	DOC-XX.XXX.XXX.XXX				
					Лист				
					30				

[illegible]

Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата

					DOC-XX.XXX.XXX.XXX	Лист
Изм.	Лист	№ докум.	Подп.	Дата		31