Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |  |
| --- | --- |
| Допускаю к защите |  |
| Руководитель |  |
|  | подпись |
|  | Л.С. Вахрушева |
|  | И.О. Фамилия |

|  |
| --- |
| Веб-редактор сценариев диалогов ассистентов |
| для платформы создания виртуальных |

наименование темы

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине

|  |  |  |
| --- | --- | --- |
| ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНЫХ КОМПЛЕКСОВ | | |
|  | 1.026.00.00 - ПЗ (номварианта - № |  |

обозначение документа

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент |  | АСУб-21-2 |  |  |  | Д.А. Шерстянников |
|  |  | шифр группы |  | подпись |  | И.О. Фамилия |
| Нормоконтроль |  |  |  |  |  | Л.С. Вахрушева |
|  |  |  |  | подпись |  | И.О. Фамилия |

Курсовой проект защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Иркутск 2024 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

ЗАДАНИЕ

НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| По курсу | Технологии разработки программных комплексов | | | |
| Студенту | Шерстянникову Д.А. | | | |
|  | (фамилия, инициалы) | | | |
| Тема проекта | | Веб-редактор сценариев диалогов для платформы | | |
| создания виртуальных ассистентов | | | | |
|  | | | | |
| Исходные данные: | | |  | |
| Разработка веб-редактора для создания сценариев диалогов для платформы создания виртуальных ассистентов (вариант №26) | | | | |
| |  |  | | --- | --- | | Рекомендуемая литература: |  |  1. Гутгарц Р.Д Проектирование автоматизированных систем обработки информации и управления: учебное пособие для академического бакалавриата. – М.: Издательство Юрайт, 2019. 2. Проектирование АСОИУ [Электронный ресурс] : методические указания по выполнению курсового проекта: 09.03.01 «Информатика и вычислительная техника» для бакалавров по специальности «Автоматизированные системы обработки информации и управления» / Иркут. нац. исслед. техн. ун-т ; сост. Р. Д. Гутгарц. - Электрон. дан. - Иркутск : ИРНИТУ, 2018 3. Проектирование информационных систем : учебник и практикум для академического бакалавриата / Д. В. Чистов, П. П. Мельников, А. В. Золотарюк, Н. Б. Ничепорук ; под общ. ред. Д. В. Чистова. – М. : Издательство Юрайт, 2016. 4. Рудинский И.Д. Технология проектирования автоматизированных систем обработки информации и управления : учебное пособие для вузов / И. Д. Рудинский. - Москва : Горячая линия - Телеком, 2011  |  | | --- | |  | | | | |  |

Графическая часть на    –       листах.

Дата выдачи задания « » сентября 2024 г.

|  |  |  |  |
| --- | --- | --- | --- |
| Задание получил |  |  | Д.А. Шерстянников |
|  | подпись |  | И.О. Фамилия |

Дата представления проекта руководителю « » декабря 2024 г.

|  |  |  |  |
| --- | --- | --- | --- |
| Руководитель курсового проектирования |  |  | Л.С. Вахрушева |
|  | подпись |  | И.О. Фамилия |

Содержание

[Введение 4](#_Toc184656476)

[1 Анализ предметной области 6](#_Toc184656477)

[1.1 Виртуальные ассистенты и их роль 6](#_Toc184656478)

[1.2 Основные компоненты виртуального ассистента 6](#_Toc184656479)

[1.3 Проблемы при создании виртуальных ассистентов 6](#_Toc184656480)

[1.4 Визуальные редакторы сценариев диалогов 7](#_Toc184656481)

[1.5 Обзор существующих программных средств 7](#_Toc184656482)

[1.5.1 Google Dialogflow 7](#_Toc184656483)

[1.5.2 Яндекс Диалоги 8](#_Toc184656484)

[1.5.3 Microsoft Bot Framework 9](#_Toc184656485)

[1.5.4 Just AI CP 9](#_Toc184656486)

[1.6 Сравнение 9](#_Toc184656487)

[2 Проектирование 11](#_Toc184656488)

[2.1 Описание бизнес-процессов 11](#_Toc184656489)

[2.1.1 Текущее состояние (AS IS) 11](#_Toc184656490)

[2.1.2 Планируемое состояние (TO BE) 12](#_Toc184656491)

[2.2 Описание вариантов использования 13](#_Toc184656492)

[2.3 Выработка требований и постановка задачи 13](#_Toc184656493)

[2.4 Выбор и обоснование средств проектирования и реализации 14](#_Toc184656494)

[2.5 Проектирование архитектуры приложения 14](#_Toc184656495)

[2.6 Проектирование хранилища данных 19](#_Toc184656496)

[2.7 Проектирование пользовательского интерфейса 22](#_Toc184656497)

[3 Разработка 27](#_Toc184656498)

[Заключение 30](#_Toc184656499)

[Список использованных источников 31](#_Toc184656500)

# Введение

Виртуальные ассистенты представляют собой набор контекстно-ориентированных сервисов, способных взаимодействовать с пользователем посредством естественного языка. Их главная функция заключается в поддержке интерактивного диалога, выполнении задач и предоставлении информации по запросу. Современные виртуальные ассистенты, такие как Google Assistant, Яндекс.Алиса и Apple Siri, играют ключевую роль в улучшении пользовательского опыта за счёт автоматизации рутинных действий и быстрого предоставления данных. Эти ассистенты способны не только выполнять команды, но и решать задачи от имени пользователя, взаимодействуя с внешними сервисами и предоставляя решения на основе анализа контекста и входных данных.

Типичная работа виртуального ассистента начинается с получения или текстового, или голосового ввода, который затем преобразуется в текстовое представление, обрабатывается алгоритмами обработки естественного языка для определения намерений пользователя и последующих действий. Эти действия могут включать поиск информации, взаимодействие с различными сервисами, управление устройствами или выполнение задач, таких как организация встреч или отправка сообщений. После выполнения задачи виртуальный ассистент генерирует ответ и предоставляет его пользователю в удобной форме.

Современные платформы для создания виртуальных ассистентов, такие как Google Dialogflow, Яндекс Диалоги и Just AI, предоставляют инструменты для создания собственных ассистентов. Они предлагают готовые решения для распознавания речи, генерации ответов, управления диалогами и интеграции с внешними сервисами. Однако создание эффективных виртуальных ассистентов требует глубокого понимания бизнес-процессов, для которых они разрабатываются, а также правильной настройки компонентов, таких как обработка естественного языка, системы управления диалогами и компоненты для принятия решений.

С ростом популярности виртуальных ассистентов и увеличением числа задач, которые они могут выполнять, возрастает потребность в инструментах, облегчающих разработку и настройку таких систем. Визуальные редакторы сценариев диалогов становятся важной частью экосистемы разработки ВА, позволяя разработчикам и бизнес-пользователям быстро создавать сложные сценарии взаимодействия без необходимости глубоких знаний в области программирования.

Виртуальные ассистенты могут быть интегрированы с различными устройствами и сервисами, такими как умные дома, автомобили и другие IoT-устройства. Это позволяет пользователям повысить удобство работы в различных отраслях, например, виртуальные ассистенты могут быть использованы для предоставления медицинской информации, управления личными финансами, или для помощи в обучении.

Целью проекта является упрощение работы при создании спецификации для виртуального помощника посредством веб-редактора сценариев для разработки виртуальных ассистентов, который обеспечит пользователям возможность быстро и эффективно создавать, и настраивать ассистентов для решения конкретных проблем.

Для достижения цели определены следующие задачи:

1. Исследовать существующие платформы и инструменты для разработки виртуальных ассистентов, такие как Google Dialogflow, Яндекс Диалоги и Just AI.
2. Проанализировать типовые сценарии взаимодействия виртуальных ассистентов и выявить ключевые компоненты для создания эффективных решений.
3. Разработать визуальный редактор, позволяющий пользователям создавать сценарии диалогов с использованием готовых блоков.
4. Обеспечить возможность интеграции с внешними сервисами для расширения функционала ассистентов.
5. Провести тестирование интерфейса редактора для обеспечения удобства использования и производительности системы.

# Анализ предметной области

# Виртуальные ассистенты и их роль

Виртуальные ассистенты представляют собой сложные программные системы, которые позволяют пользователям взаимодействовать с устройствами или сервисами на естественном языке. В отличие от традиционных интерфейсов, где пользователю требуется вручную вводить команды или нажимать кнопки, ВА могут интерпретировать запросы, которые приходят в текстовой или голосовой форме, и реагировать на них в режиме реального времени.

Существует множество примеров использования виртуальных ассистентов в различных сферах. В бытовых приложениях ассистенты могут управлять умными устройствами (освещение, климат-контроль), предоставлять информацию (погода, новости) и выполнять задачи, такие как отправка сообщений или организация встреч. В корпоративной среде ВА могут использоваться для автоматизации рабочих процессов, помощи клиентам, поддержки сотрудников и интеграции с бизнес-системами.

Главная особенность ВА заключается в их способности «понимать» пользователя, а также адаптироваться к различным контекстам и условиям. Например, если пользователь запросит информацию о встрече, ассистент может не только найти соответствующие данные в календаре, но и предложить напомнить о встрече или связаться с участниками. Эта контекстная адаптивность требует использования сложных алгоритмов для распознавания и интерпретации намерений пользователя.

# Основные компоненты виртуального ассистента

Виртуальные ассистенты включают в себя несколько ключевых компонентов:

1. Компоненты обработки естественного языка. Эти компоненты отвечают за распознавание речи, преобразование голоса в текст и обратный процесс – генерацию голосового ответа. Также они занимаются интерпретацией текста для понимания намерений пользователя и определения дальнейших действий.
2. Система управления диалогом. Этот компонент определяет логику взаимодействия между пользователем и ассистентом. Он отслеживает контекст разговора, выбирает подходящие ответы и корректирует диалог в зависимости от намерений пользователя.
3. Компоненты принятия решений. Включают алгоритмы, которые анализируют запросы пользователя и определяют, как выполнить задачу. Они могут интегрироваться с внешними сервисами, базами данных и другими системами для получения информации и выполнения действий.
4. Интеграционные компоненты. Они отвечают за взаимодействие ассистента с другими сервисами, такими как умные устройства, системы управления задачами, CRM-системы и другие внешние платформы.

# Проблемы при создании виртуальных ассистентов

Несмотря на успехи в развитии технологий виртуальных ассистентов, их разработка остаётся сложной задачей, которая требует глубокого понимания различных аспектов, начиная от обработки естественного языка и заканчивая интеграцией с внешними сервисами. Одна из основных проблем заключается в разнообразии задач, для которых создаются ассистенты. Каждая задача требует создания специфической логики и сценариев взаимодействия, что может сделать процесс разработки трудоёмким и сложным.

Ещё одной проблемой является обеспечение качества взаимодействия. Виртуальные ассистенты должны быть способны не только точно интерпретировать запросы пользователя, но и предлагать решения, соответствующие ожиданиям. Для этого требуется правильная настройка всех компонентов ассистента.

# Визуальные редакторы сценариев диалогов

Для упрощения разработки ВА были созданы визуальные редакторы сценариев, которые позволяют пользователям строить диалоги с использованием графического интерфейса. В таких редакторах пользователи могут выбирать готовые блоки для выполнения различных задач: интерпретации запросов, обработки данных, генерации ответов. Это значительно упрощает процесс создания ассистентов и делает его доступным даже для тех, кто не обладает глубокими знаниями в области программирования.

# Обзор существующих программных средств

Современные платформы для создания виртуальных ассистентов предлагают широкий спектр инструментов и решений. Сейчас есть несколько популярных платформ, которые предлагают готовые решения для разработки систем взаимодействия с пользователями. Например, Google Dialogflow, Яндекс Диалоги, Microsoft Bot Framework и Just AI. Каждая из этих платформ имеет свои плюсы и минусы, которые нужно учитывать при выборе инструмента для конкретной задачи.

Например, Google Dialogflow позволяет создавать сложные диалоговые сценарии и интегрироваться с другими сервисами Google. Яндекс Диалоги предлагает мощные инструменты для анализа естественного языка и интеграцию с другими сервисами Яндекса. Microsoft Bot Framework позволяет создавать ботов для различных платформ, включая Skype и Slack. Just AI предлагает простой и интуитивно понятный интерфейс для создания виртуальных ассистентов. Каждая из этих платформ имеет свои особенности, преимущества и ограничения, которые влияют на выбор инструментов для решения конкретных задач.

# Google Dialogflow

Google Dialogflow – это облачная платформа для разработки чат-ботов и виртуальных ассистентов, предоставляющая инструменты для обработки естественного языка и управления диалогами. Dialogflow активно используется для создания ассистентов, которые могут интегрироваться с различными системами, такими как Google Assistant, Facebook Messenger, Slack, а также с пользовательскими веб-сайтами, мобильными приложениями, устройствами IoT.

Dialogflow может интерпретировать намерения пользователей и извлекать сущности из их запросов, что позволяет создавать более естественные взаимодействия.

Преимущества:

* интегрируется с экосистемой Google, включая Google Assistant и облачные сервисы, что делает её удобной для пользователей, работающих в этой среде;
* поддерживает большое количество языков, что упрощает разработку многоязычных решений;
* автоматически обучает модели на основе входных данных, улучшая точность распознавания и интерпретации запросов пользователя;
* интуитивно понятный редактор позволяет создавать и редактировать сценарии диалогов без необходимости программирования;
* есть возможность создания виртуальных ассистентов для устройств интернета вещей.

Недостатки:

* создание сложных сценариев и интеграция с нестандартными системами могут требовать значительных технических навыков;
* платформа лучше всего работает с сервисами Google, и интеграция с внешними системами может быть ограничена;
* графический редактор не очень удобен для сложных задач.

# Яндекс Диалоги

Яндекс Диалоги – это платформа для создания чат-ботов и виртуальных ассистентов, разработанная компанией Яндекс. Платформа особенно популярна в русскоязычных странах и поддерживает интеграцию с сервисом Яндекс.Алиса, который является одним из ведущих виртуальных помощников в России.

Преимущества:

* платформа позволяет легко подключать чат-ботов и ассистентов к продуктам Яндекс;
* предоставляет инструменты для качественной обработки русского языка, что делает её одной из лучших платформ для создания русскоязычных ассистентов;
* предоставляет удобный редактор для создания сценариев взаимодействия, который не требует глубоких технических знаний.

Недостатки:

* в отличие от Dialogflow, платформа менее приспособлена для работы с многими иностранными языками, что делает её менее универсальной для международных проектов;
* платформа ориентирована на продукты и услуги Яндекса, что может ограничить её использование в других регионах;
* для создания сложных сценариев возможностей платформы может быть недостаточно.

# Microsoft Bot Framework

Microsoft Bot Framework – это платформа для разработки виртуальных ассистентов и чат-ботов от компании Microsoft. Платформа предоставляет инструменты для создания сложных ботов с поддержкой мультимодальных интерфейсов, интеграции с различными каналами общения и поддержки естественного языка.

Преимущества:

* предоставляет широкие возможности для настойки и интеграции с различными системами, используя языки программирования, что делает его подходящим для создания сложных корпоративных решений;
* поддерживает интеграцию с Microsoft Teams, Skype, Facebook Messenger, Slack и другими популярными каналами связи;
* имеет встроенные возможности для обработки речи и текста с использованием Microsoft Azure Cognitive Services;

Недостатки:

* требует значительных технических знаний для полной настройки и использования всех возможностей;
* использование Microsoft Azure для интеграции и обработки данных является дорогим решением, особенно для небольших проектов;
* сильно зависит от экосистемы Microsoft.

# Just AI CP

JAICP (Just AI Conversational Platform) – это российская платформа для разработки чат-ботов и виртуальных ассистентов, которая активно используется для создания ассистентов с поддержкой русского языка и интеграцией с различными популярными мессенджерами и голосовыми помощниками.

Преимущества:

* позволяет создавать как простые, так и сложные сценарии взаимодействия, включая мультимодальные интерфейсы и поддержку голосовых команд;
* имеет визуальный редактор и собственный язык сценариев;
* поддерживает интеграцию с Telegram, WhatsApp, Viber, а также с российскими сервисами голосовых помощников;
* хорошо справляется с обработкой русского языка, что делает её подходящей для проектов, ориентированных на русскоязычных пользователей.

Недостатки:

* для более сложных решений могут потребоваться дополнительные интеграции и доработки.

# Сравнение

Итоговое сравнение аналогичных решений представлено в таблице 1.1.

Таблица 1.1 – Сравнение аналогичных решений

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерий | Dialogflow | Яндекс.Диалоги | Microsoft Bot Framework | Just AI |
| Поддержка языков | Многоязычная поддержка, основные мировые языки | Фокус на русский язык, слабая поддержка иностранных языков | Широкая поддержка языков через Azure Cognitive Services | Фокус на русский язык, слабая поддержка иностранных языков |
| Интеграции | Интеграция с Google сервисами, популярными мессенджерами и IoT | Интеграция с сервисами Яндекса, популярными сервисами | Возможности интеграции с Microsoft Teams, Slack и др | Интеграция с мессенджерами и голосовыми помощниками |
| Машинное обучение | Встроенное обучение моделей на входных данных | Ограниченные возможности обучениям | Продвинутые возможности с использованием Azure Cognitive Services | Поддерживает обучение, но ограничено по сравнению с конкурентами |
| Гибкость сценариев | Поддержка сложных диалогов, но требует программирования | Простые сценарии без программирования | Высокая гибкость, требует программирования | Поддержка любых сценариев через собственный язык |
| Простота использования | Интуитивно понятный интерфейс для простых задач | Удобен для создания простых ботов | Сложный интерфейс, требует технических знаний | Визуальный редактор, не требует глубоких знаний |

Сравнение существующих платформ показывает, что каждое решение имеет сильные и слабые стороны, связанные с интеграциями и сложностью использования. Все платформы в основном ориентированы на использование в собственной экосистеме. Также во всех приложениях много ручной работы. Приходится каждый раз делать много типовых действий при создании сценария работы. Ценность предлагаемой разработки заключается в создании редактора, который позволит автоматизировать большинство рутинных действий при создании сценария с помощью удобного визуального редактора с блоками.

# Проектирование

# Описание бизнес-процессов

В настоящее время создание виртуального ассистента сопряжено с определёнными трудностями и неудобствами. Различные платформы предлагают пользователям базовые инструменты, такие как условные ветвления, блоки кода и возможности текстового ввода. Однако, чтобы эффективно использовать эти инструменты для разработки помощников, требуется значительное время и знания в области программирования.

Кроме того, необходимо тщательно продумать концепцию виртуального ассистента, изучить предметную область, сформулировать задачи и разработать модели данных, которые будут необходимы для работы.

# Текущее состояние (AS IS)

В общем процесс создания помощника с существующими решениями выглядит так: формируются требования, составляются модели данных, создаётся сценарий и пишется код, собирается спецификация. На рисунках 3.1 и 3.2 показан процесс и его декомпозиция соответственно.

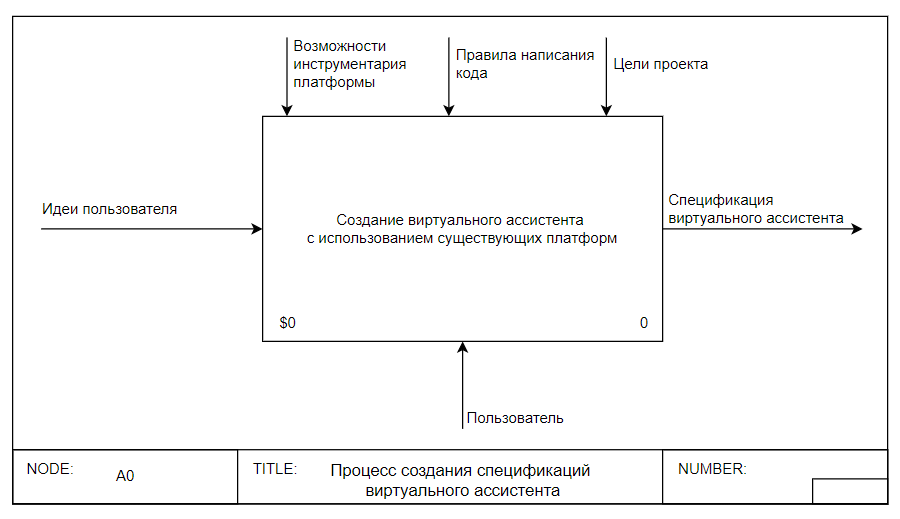


Рисунок 3.1 – Контекстная диаграмма AS IS

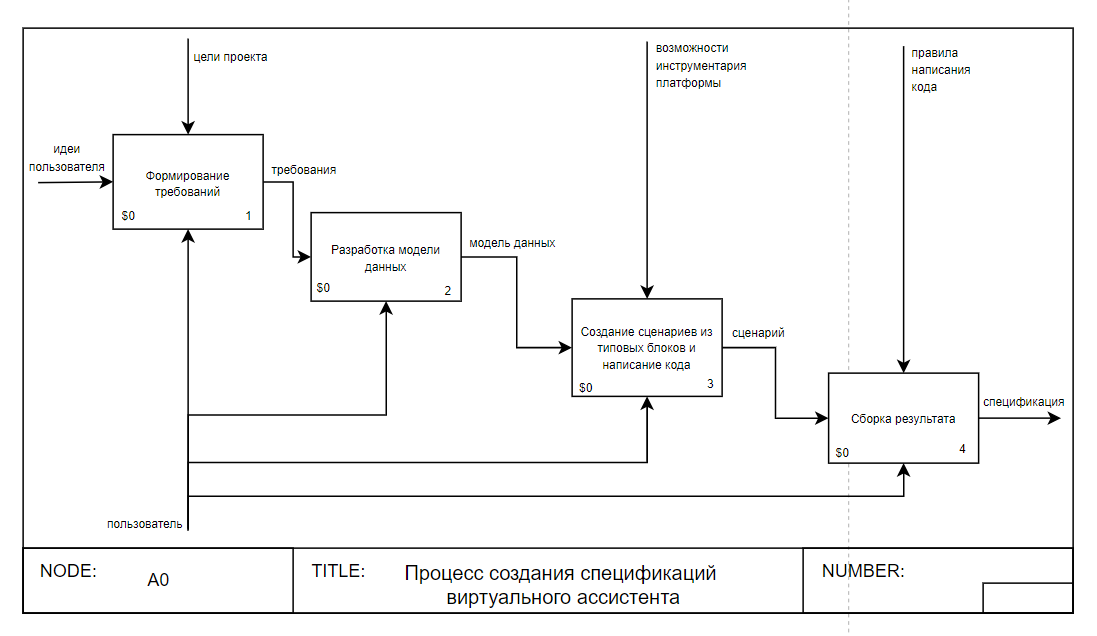


Рисунок 3.2 – Декомпозиция диаграммы A0 AS IS

# Планируемое состояние (TO BE)

В целевом бизнес-процессе предусматривается оптимизация процесса создания виртуального ассистента, где упор делается на минимизацию шаблонных действий и увеличение удобства для пользователей. Предполагается создание специальных блоков, ориентированных на решение предопределённых классов задач, а также заранее продуманные модели данных. Это позволяет значительно упростить затраты времени пользователя во время разработки, а также убирает необходимость долгой настройки схемы и написания кода для некоторых блоков. Блоки для решения классов проблем уже приспособлены к моделям данных, остаётся только настроить их под нужды пользователя. На рисунках 3.3 и 3.4 показан целевой процесс и его декомпозиция соответственно.

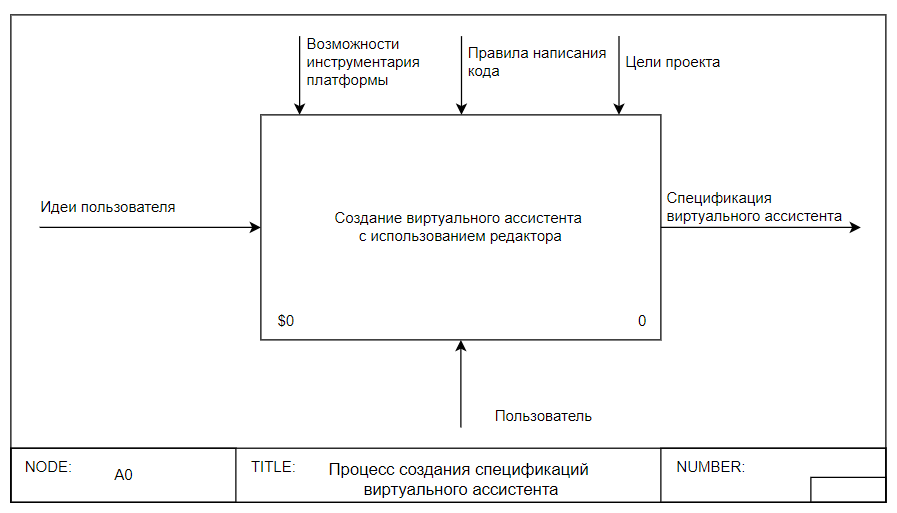
****

Рисунок 3.3 – Контекстная диаграмма TO BE

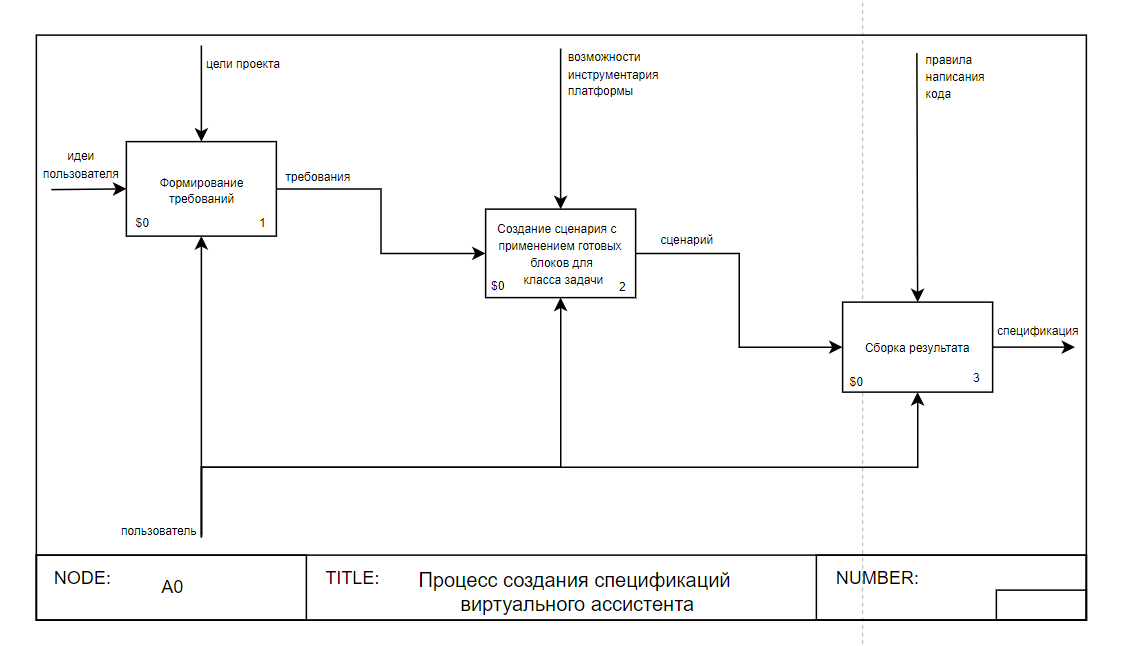
****

Рисунок 3.4 – Декомпозиция диаграммы A0 TO BE

# Описание вариантов использования

Основным пользователем редактора является проектировщик виртуальных ассистентов, который составляет из предоставленных ему средств сценарий работы виртуального помощника. Проектировщику требуется понимание предметной области и задач помощника. Диаграмма вариантов использования показана на рисунке 4.1.

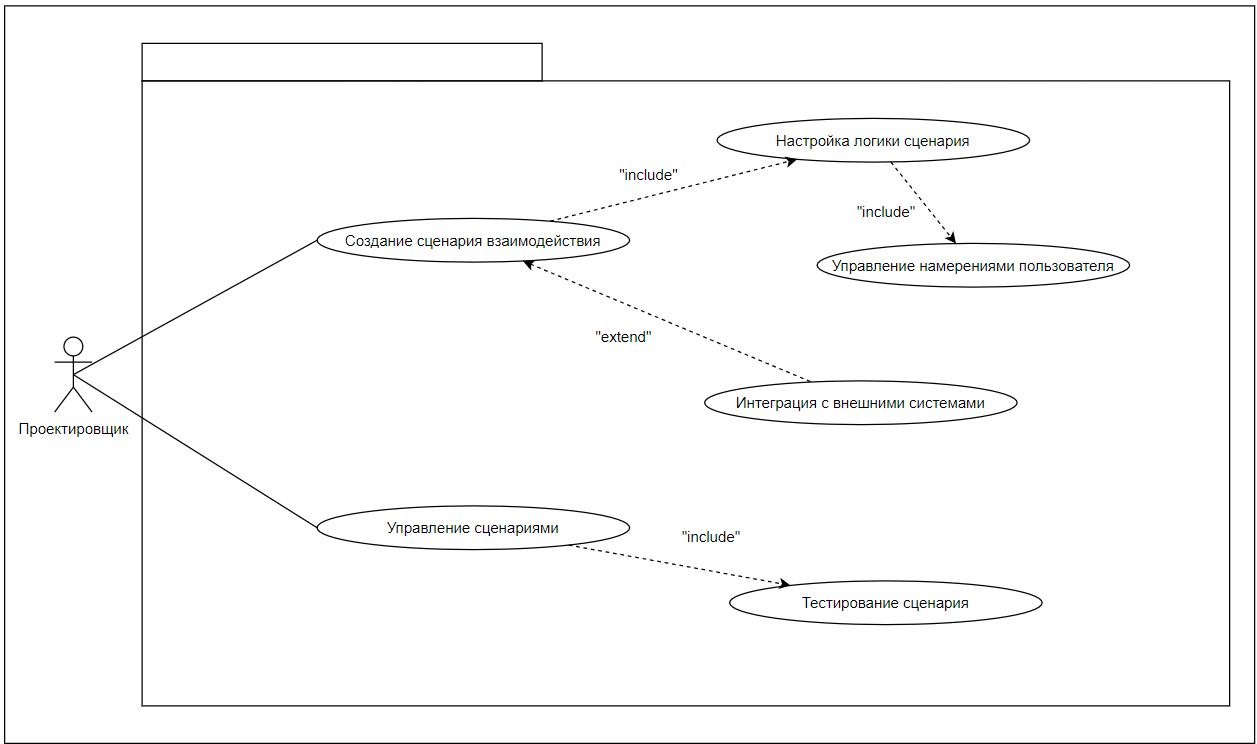


Рисунок 4.1 – Диаграмма варианта использования проектировщика

# Выработка требований и постановка задачи

Цель – создать визуальный редактор, который позволит пользователю создать сценарий работы виртуального ассистента и получить спецификацию. Редактор должен позволить пользователям с разными знаниями в области программирования и моделирования данных создавать сценарии разной сложности посредством типовых и специализированных блоков.

Для этого нужно решить следующие задачи:

* разработать визуальный редактор, который позволит пользователю создать сценарий работы виртуального ассистента;
* реализовать поддержку создания сложных сценариев работы виртуального ассистента;
* разработать систему автоматической генерации спецификации на основе созданного сценария;
* реализовать экспорт спецификации в различных форматах;
* протестировать визуальный редактор на предмет его функциональности и пользовательского интерфейса.

# Выбор и обоснование средств проектирования и реализации

Для проектирования были выбраны следующие средства:

* UML (англ. Unified Modeling Language – унифицированный язык моделирования) – это стандартный язык спецификации, визуализации и построения автоматизированной системы, применяемый в сферах, связанных с разработкой программного обеспечения;
* Модель C4 (от англ. С4 model, Context Container Component Code model, модель «контекст-контейнер-компонент-код») – простой метод графической записи для моделирования архитектуры программных систем. Он основан на структурной декомпозиции системы на контейнеры и компоненты и опирается на существующие методы моделирования, такие как UML, для более детальной декомпозиции архитектурных блоков;
* Draw.io – программа для создания любого типа диаграмм, включая UML и C4. Имеет встроенные шаблоны, элементы и инструменты, которые потребуются для создания необходимых схем.

Для разработки были выбраны следующие средства:

* Python – это высокоуровневый язык программирования который обеспечивает быструю разработку и предоставляет широкие возможности для создания различных функций приложения;
* JavaScript – это язык программирования, который используется для создания интерактивных веб-страниц и приложений;
* React.js – это библиотека JavaScript для создания пользовательских интерфейсов. Она обеспечивает быструю разработку и предоставляет много возможностей для создания различных функций приложения. Уже используется в проекте, поэтому упростит взаимодействие с другими командами;
* ReactFlow – это библиотека для React, которая предоставляет возможности создания различных интерактивных холстов разной сложности;
* Adobe Spectrum – это библиотека настраиваемых компонентов для React, которая позволяет создавать различные пользовательские интерфейсы;
* FastAPI – это простой и мощный инструмент для создания API и серверных функций. Также уже используется в проекте и упростить интеграцию;
* Visual Studio Code – это интегрированная среда разработки, которая используется для написания и редактирования кода. Благодаря поддержке множества расширений, она позволяет ускорить разработку проекта.

# Проектирование архитектуры приложения

Тщательное проектирование архитектуры важно для разработки удобного и функционального приложения. Разрабатываемый редактор предназначен для пользователей, которым нужно создать спецификацию для виртуального ассистента. Оно интегрируется с основной платформой, получая возможность идентификации учётных записей пользователей и работы с проектами.

Архитектура приложения устроена в виде веб-редактора, который взаимодействует с основной платформой, чтобы получать данные, необходимые для работы. На рисунке 7.1 показана диаграмма контекста в нотации С4, отображающая взаимодействие пользователя, системы и её компонентов. Пользователь с помощью веб-редактора проектирует сценарии голосового помощника, а веб-редактор, в свою очередь, обменивается данными с системой управления процессом разработки.

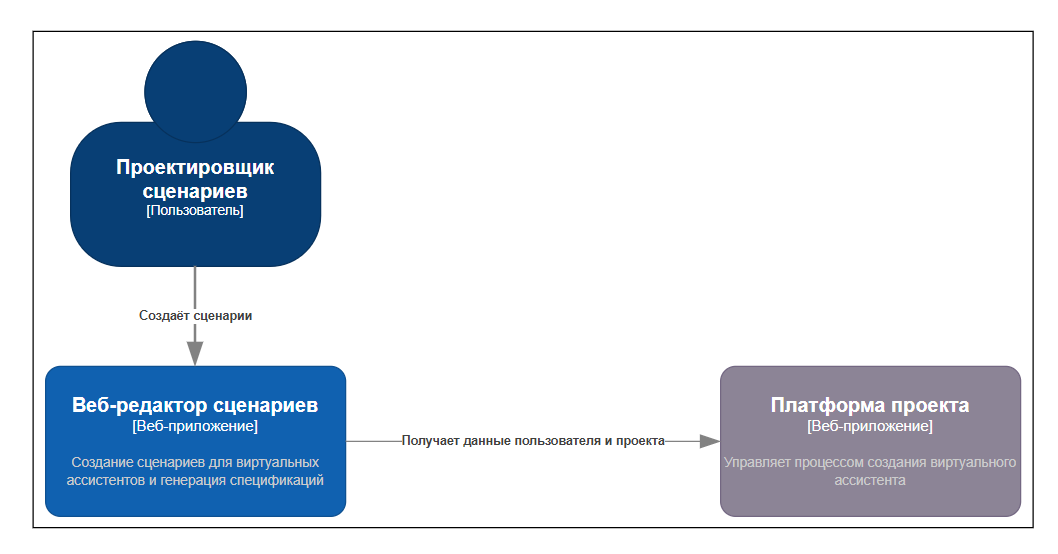


Рисунок 7.1 – Диаграмма контекста

Внутреннее устройство приложения включает клиентскую часть, реализованную с использованием библиотеки React.js. Эта часть обеспечивает интерфейс для проектирования сценариев и взаимодействия с визуальными элементами. Для манипуляции визуальными блоками используется библиотека ReactFlow, позволяющая размещать элементы на холсте, перемещать их и задавать взаимосвязи.

Основной язык программирования для создания спецификаций и генерации кода – JavaScript. Он обеспечивает выполнение логики взаимодействия между блоками сценариев.

Серверная часть приложения реализована на основе фреймворка FastAPI с использованием языка программирования Python. Сервер предоставляет доступ с REST API, через который клиентское приложение взаимодействует с другими сервисами платформы и базой данных.

На рисунке 7.2 представлена диаграмма контейнера веб-редактора, в которой отображены основные компоненты системы: клиентская часть, сервер и база данных. Клиент предоставляет пользователю инструменты для проектирования сценариев, сервер обрабатывает запросы от клиента, а база данных используется для хранения данных проектов, пользовательских сведений и информации, отображаемой в блоках редактора.

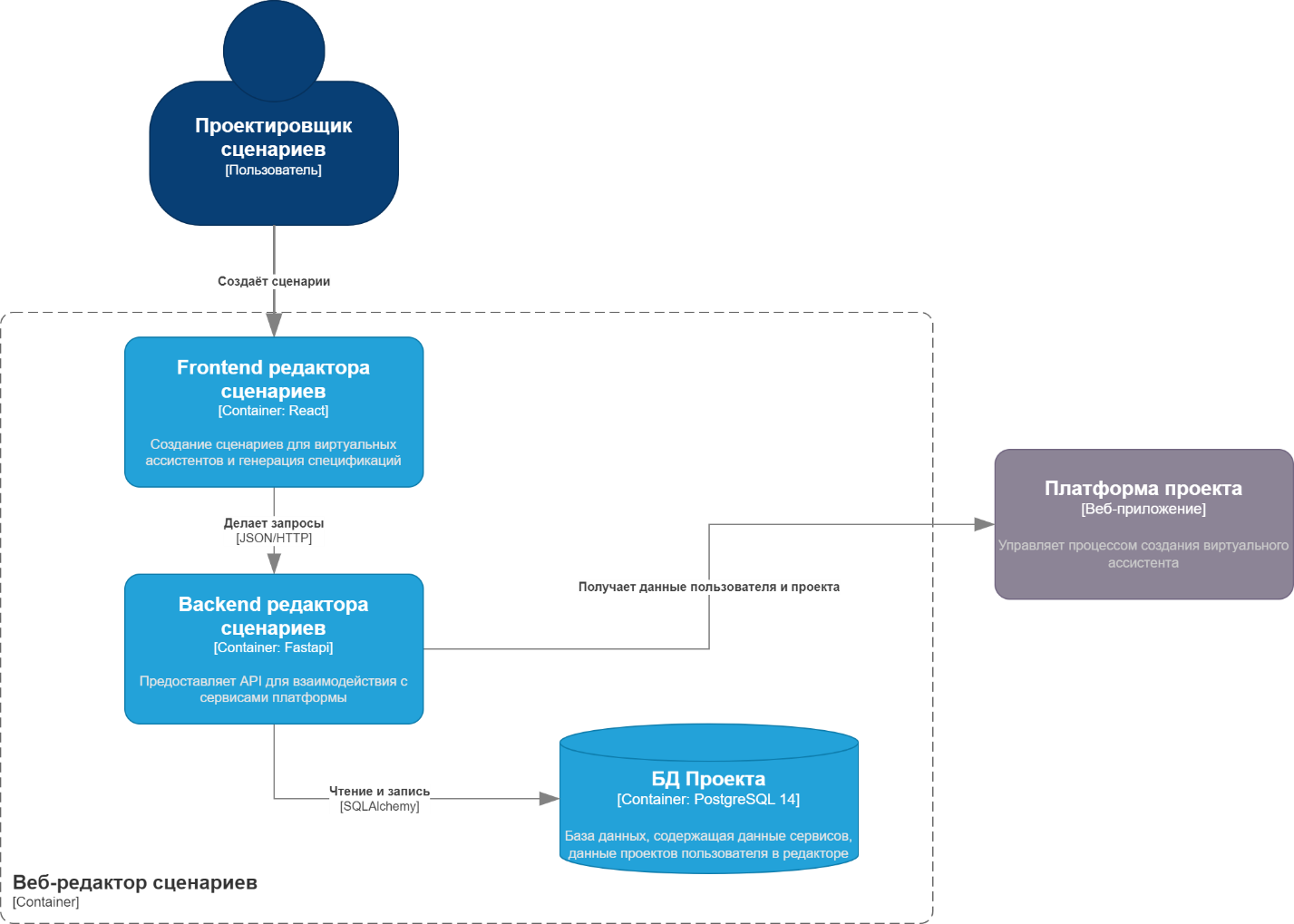


Рисунок 7.2 – Диаграмма контейнеров веб-редактора

На рисунке 7.3 изображена диаграмма компонентов серверной части приложения. Сервер включает модули маршрутизации запросов, взаимодействия с базой данных и интеграции с другими приложениями платформы. Также предусмотрены механизмы преобразования данных, необходимых для работы редактора.

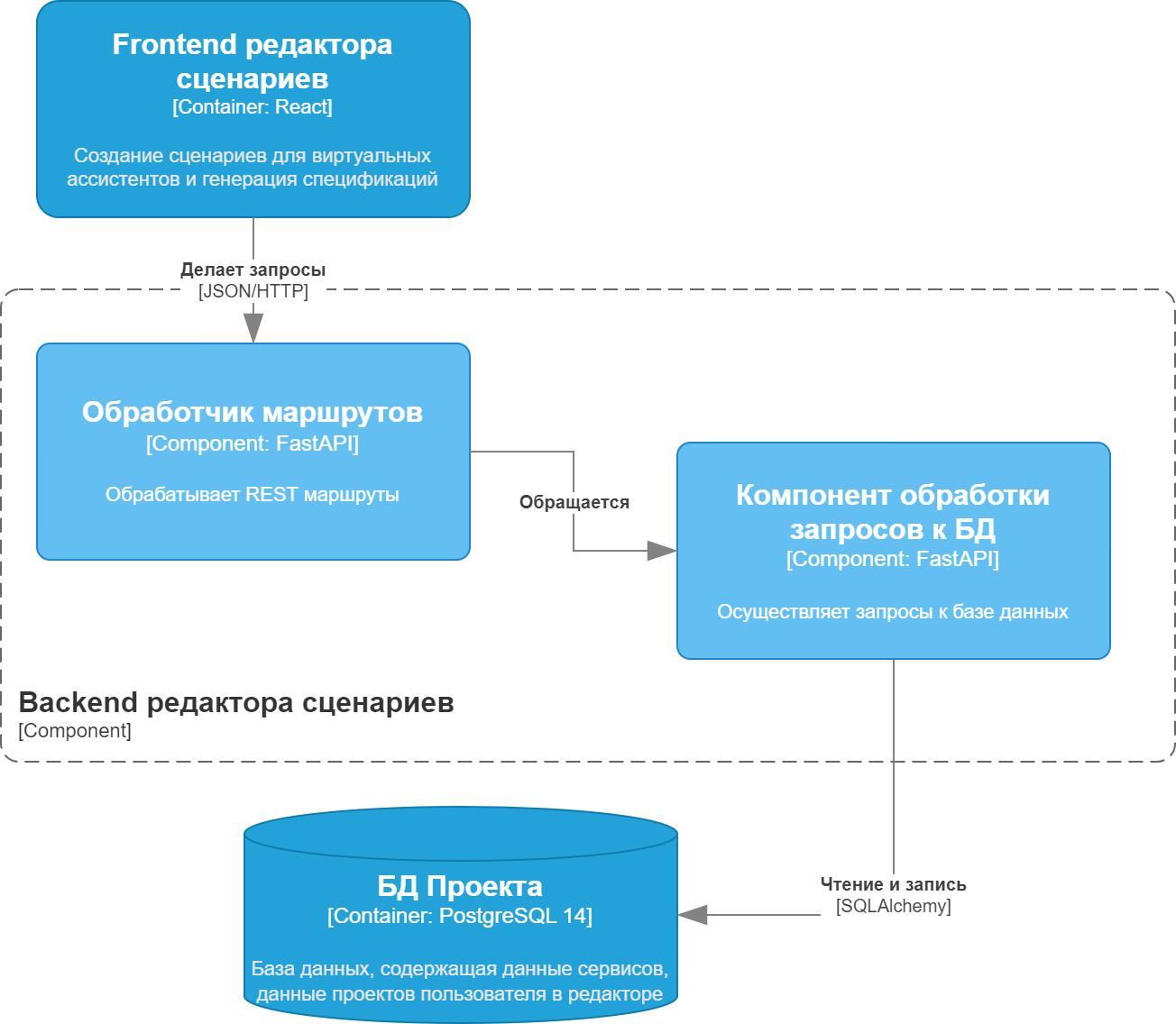


Рисунок 7.3 – Диаграмма компонентов сервера приложения

На рисунке 7.4 показана диаграмма компонентов интерфейса приложения. Основные элементы интерфейса включают меню, предоставляющее пользователю доступ к функциям редактора, и холст, где пользователь проектирует сценарии. Для генерации спецификаций используется компонент генератора, который анализирует структуру сценариев, определяет последовательность действий и формирует текстовое описание каждого блока в формате, подходящем для последующей обработки.

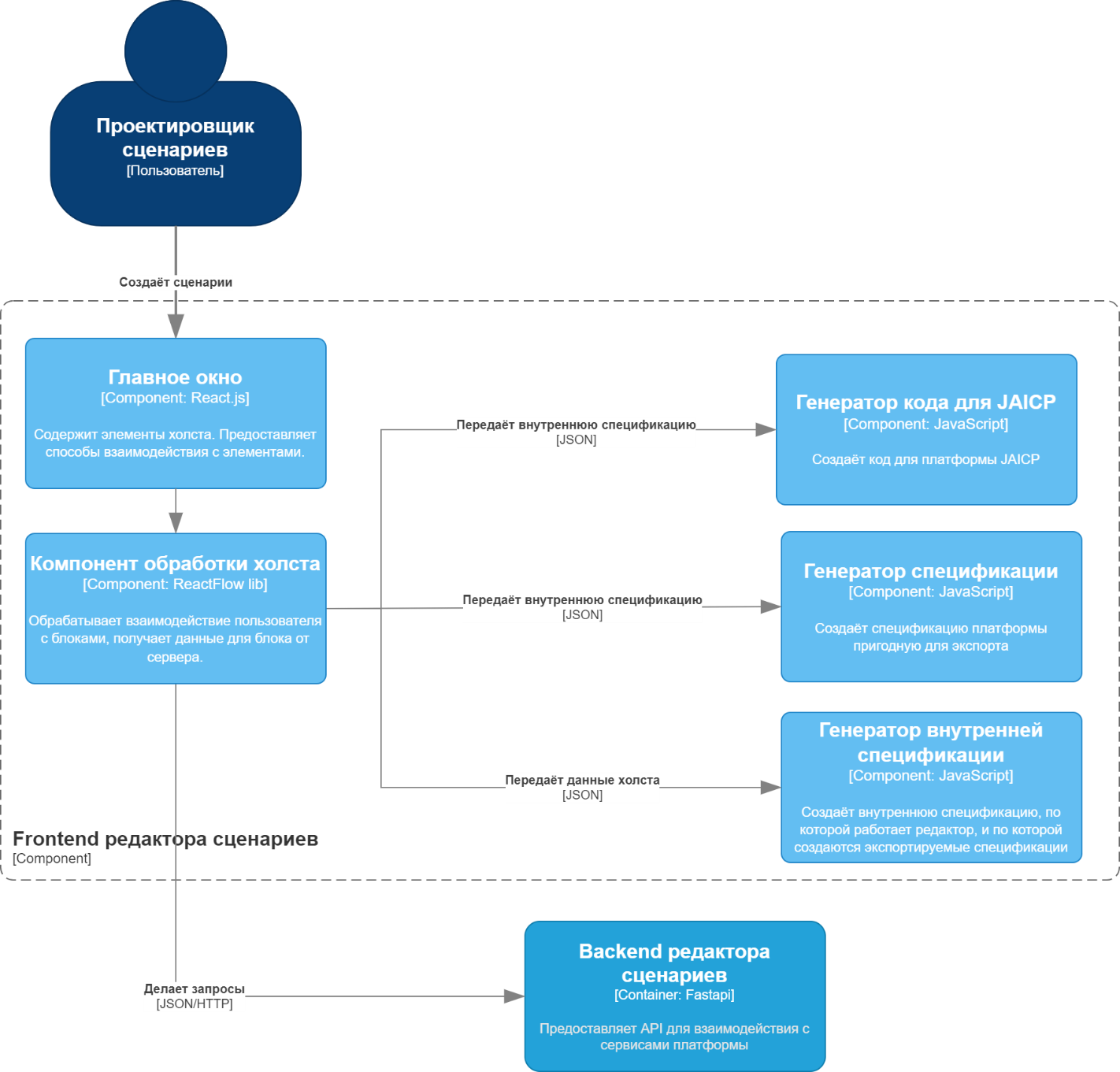


Рисунок 7.4 – Диаграмма компонентов интерфейса приложения

Данная архитектура обеспечивает гибкость, масштабируемость и удобство работы с приложением. Она позволяет пользователю сосредоточиться на проектировании сценариев, предоставляя все необходимые инструменты для интеграции и обработки данных.

# Проектирование хранилища данных

Для данного проекта не требуется хранения большого числа данных, так как его основные задачи заключаются в интерактивном взаимодействии с пользователем и в генерации новых данных, а вся необходимая информация собирается из уже реализованной базы данных. Однако для удобства использования необходимо сохранять состояние сценариев, чтобы переключаться между ними и возобновлять работу после перерыва.

Основной сущностью хранилища является **проект**, представляющий состояние визуального редактора для конкретного пользователя. Проекты хранятся в отдельной таблице, структура которой представлена в таблице 2.1.

Таблица 2.1 – Описание таблицы для хранения проектов

|  |  |  |
| --- | --- | --- |
| № | Атрибут | Описание |
| 1 | id | уникальный идентификатор записи |
| 2 | name | название проекта |
| 3 | description | описание проекта |
| 4 | author\_id | идентификатор пользователя |
| 5 | chart\_data | объект с данными о состоянии всех расположенных на холсте элементов |

Для работы редактора используются данные о компонентах и их функциях. Эти данные нужны, чтобы заполнить блоки внешних сервисов. Таблица компонентов описывает отдельные элементы, используемые в сценариях. Таблица функций связывает компоненты с действиями, которые они могут выполнять, и таблица сервисов содержит данные о внешних сервисах, которые используются для выполнения функций.

На рисунке 8.1 показаны связи таблиц, которые описывают компоненты и функции. Компонент, например, «работа с погодой», может содержать функции для получения данных – прогноз, суммарная информация за период и т.д. Для выполнения функции может быть использован один из предложенных сервисов, обладающих свойствами и параметрами.

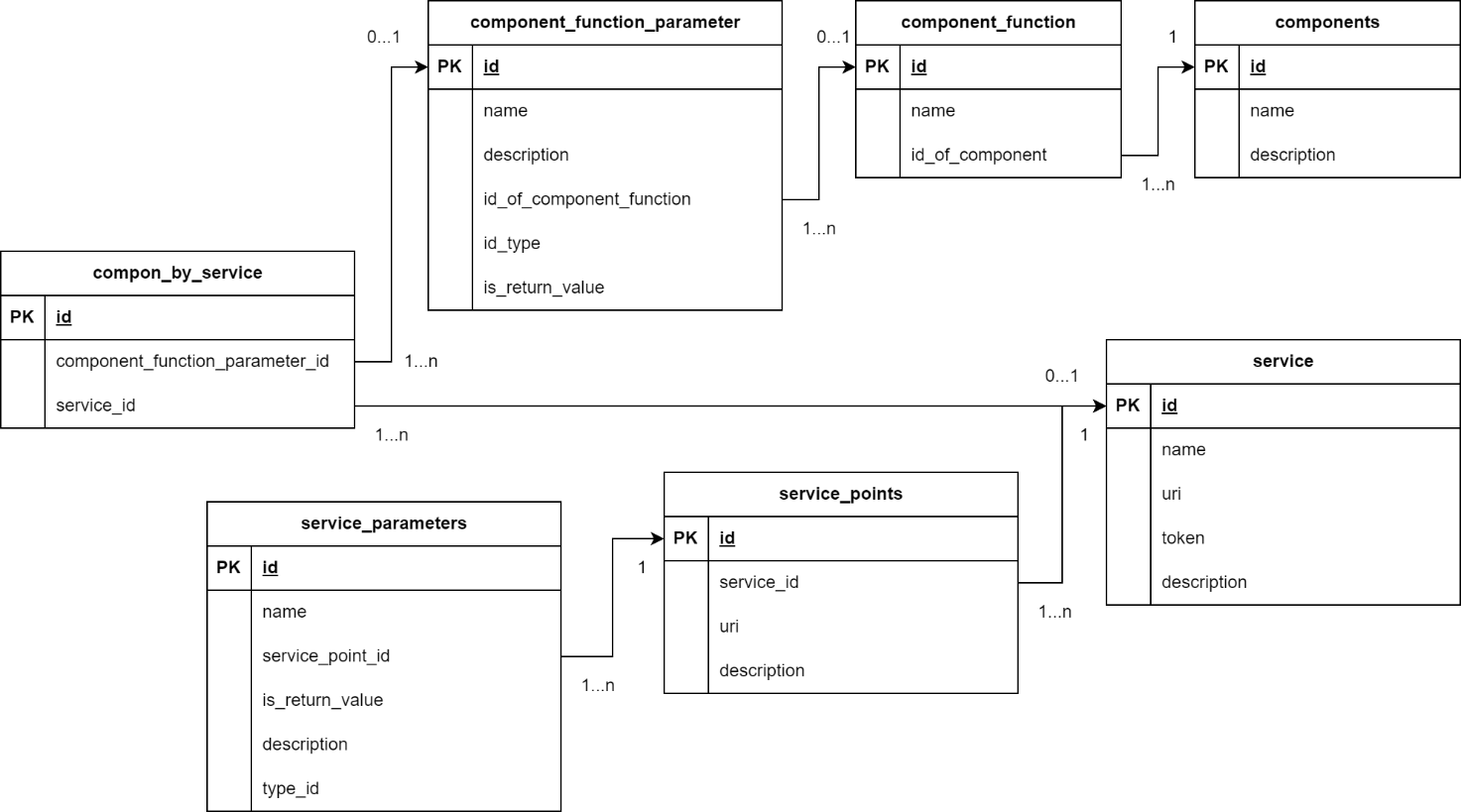


Рисунок 8.1 – Связи таблиц сервисов и компонентов

Данные из таблиц получаются при помощи запросов SQL. Один из примеров запроса для получения параметров функций показан на рисунке 8.2.

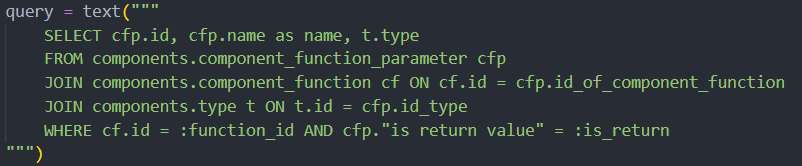


Рисунок 8.2 – Запрос для получения параметров функций

Редактор для работы с блоками использует объекты, которые можно представить в виде JSON-шаблона. Шаблон содержит описание компонента: положение на холсте, номер, название, компонент, функции и другая информация. На рисунке 8.3 показан шаблон данных, которые приложение формирует, чтобы пользователь увидел блок функции компонента. Массивы, отмеченные на рисунке квадратными скобками, обозначают, что они могут содержать более одного объекта.



Рисунок 8.3 – Пример сформированных данных описания блока

На рисунке 8.4 показан пример блока, который представляет компоненты.



Рисунок 8.4 – Пример кода описания блока функции компонента

# Проектирование пользовательского интерфейса

Главное окно приложения содержит следующие основные части: меню с блоками, область холста и панель спецификации. Панель спецификации расположена справа и появляется при нажатии на кнопку в верхней части экрана. Меню с блоками расположено слева и имеет подразделы для лучшей организации. В центральной части расположен холст, на который размещаются все элементы. На рисунке 9.1 показан макет главной страницы.

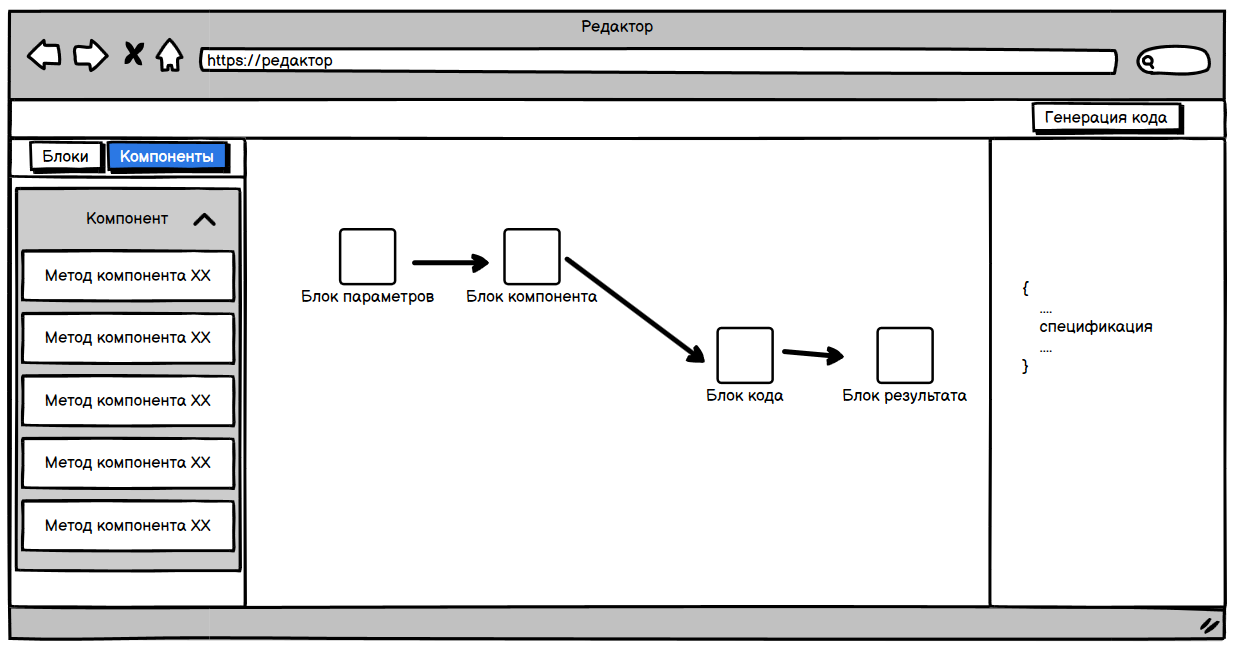


Рисунок 9.1 – Макет главной страницы

Иерархическая структура компонентов показана на рисунке 9.2. На этой схеме видно, что наибольшее число элементов содержит меню блоков.



Рисунок 9.2 – Схема экранов

Каждый блок создаётся программно и не имеет фиксированных полей ввода. В зависимости от типа данных изменяются требования к полю. Более подробно компоненты описаны в таблице 2.2. Требования к типам данных описаны в таблице 2.3.

Таблица 2.2 – Описание компонентов

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название | Поля | Описание |
| 1 | Главное окно | – | Редактор сценариев виртуального ассистента. |
| 1.1 | Меню с блоками | – | Часть главного экрана, расположенная слева. Список типовых блоков и компонентов. |
| 1.1.1 | Блок с параметрами | Значение параметра ограничивается выбранным типом данных. | Блок, в котором можно создать параметры для передачи на вход другого блока. |
| 1.1.2 | Блок условий | Входные параметры выбираются | Блок, который принимает на вход параметры из других блоков и позволяет сделать ветвление по условию. |
| 1.1.3 | Блок с кодом | Значение выбирается из входящих параметров. | Блок, в котором пользователь может производить манипуляции над данными, используя язык программирования. |
| 1.1.4 | Блок результатов | Значение выбирается из входящих параметров. | Блок, в котором определяются результирующие данные в сценарии |
| 1.1.5 | Блок авторизации | – | Блок, который используется в компонентах, если требуется авторизация в сервисах. |
| 1.1.6 | Блок компонента | Значение выбирается из входящих параметров. | Блок, в котором происходит работа с внешними сервисами. |
| 1.2 | Область холста | – | Центральная часть главного экрана, на которой пользователь взаимодействует с блоками. |
| 1.3 | Панель сгенерированного кода | – | Часть главного экрана, расположенная справа. Отображает пользователю сгенерированный код. |

Таблица 2.3 – Описание типов данных

|  |  |  |
| --- | --- | --- |
| № | Тип данных | Требования |
| 1 | int | Только целые числа |
| 2 | number | Целые числа и числа с плавающей точкой |
| 3 | string | Строки |
| 4 | bool | Булевы значения |
| 5 | struct | Сложные структуры данных |
| 6 | date | Даты |

На рисунке 9.3 представлен макет блока с параметрами. Этот блок позволяет создавать нужное пользователю число параметров, выбирать для них тип данных и задавать их значение и название. Имеет одну точку соединения на выходе.

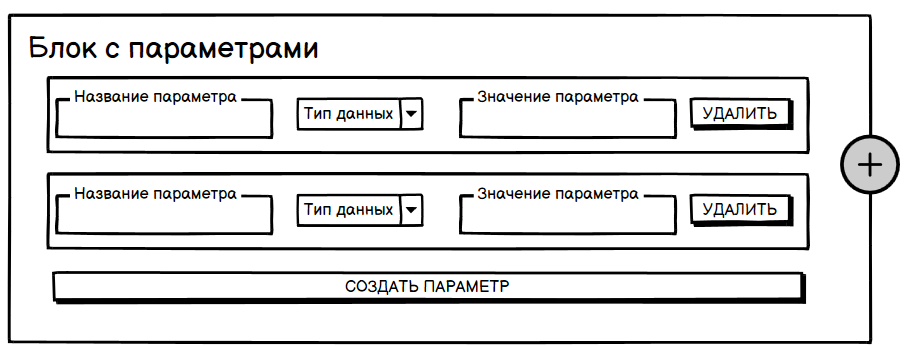


Рисунок 9.3 – Макет блока с параметрами

Блок условия может иметь несколько входящих соединений. С помощью списков выбираются параметры и условие для сравнения. Условия сравнения следующие: больше, меньше, больше или равно, больше и равно, меньше и равно, меньше или равно, равно, не равно. Блок имеет два выхода, которые нужны для определения направления в зависимости от истинности условия. Макет показан на рисунке 9.4.

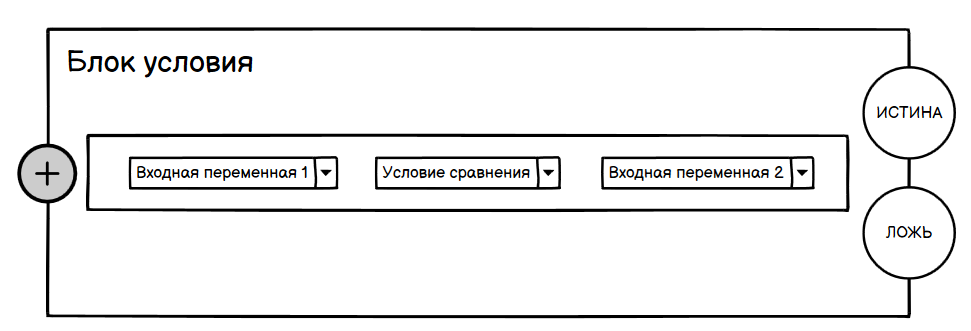


Рисунок 9.4 – Макет блока условия

Блок с кодом показан на рисунке 9.5. В верхней части создаются переменные. Названия нужны для использования в коде (центральная часть макета), значения берутся из входных параметров. Выходные параметры выбирают параметры из кода.

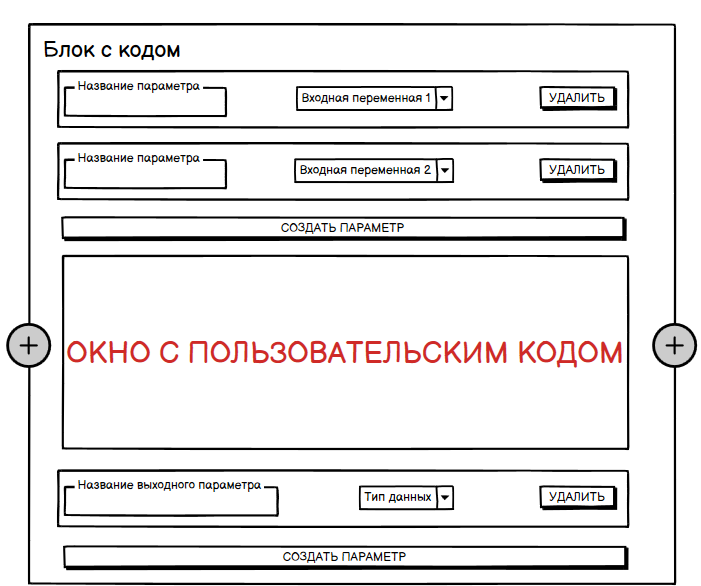


Рисунок 9.6 – Макет блока с кодом

Блок с результатами получает на вход соединения и выбирает параметры, которые нужны как результат сценария. Блок показан на рисунке 9.7.

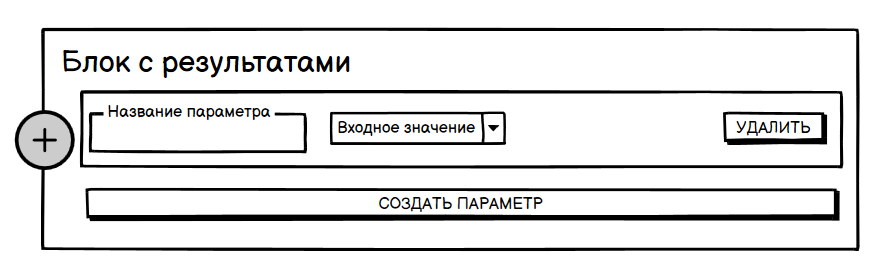


Рисунок 9.7 – Макет блока с результатами

На рисунке 9.8 показан блок авторизации. В верхней части отображается ссылка на сервис для авторизации. Ниже поля с учётными данными.

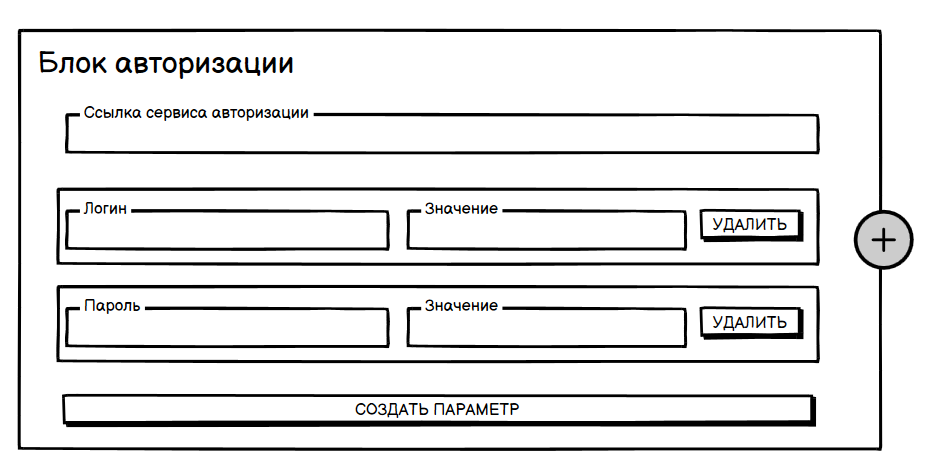


Рисунок 9.8 – Макет блока авторизации

Блок компонента имеет заданные входные и выходные параметры. Для входных данных значение выбирается из входящих соединений. В зависимости от блока, может не быть выбора сервисов. Тогда всё будет происходить автоматически. Макет блока показан на рисунке 9.9.

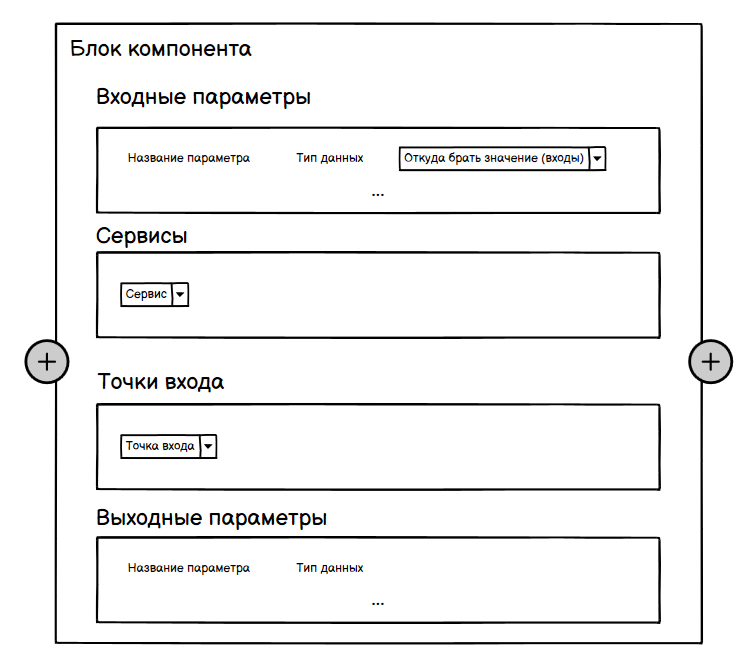


Рисунок 9.9 – Макет блока компонента

# Разработка

В ходе разработки был создан пользовательский интерфейс и серверная часть приложения.

На рисунке 3.1 представлен пример интерфейса, показывающий простую схему соединений и задание параметров. Для обеспечения корректной обработки цепочек определяются зависимые блоки, которые позволяют сохранить правильный порядок выполнения операций.

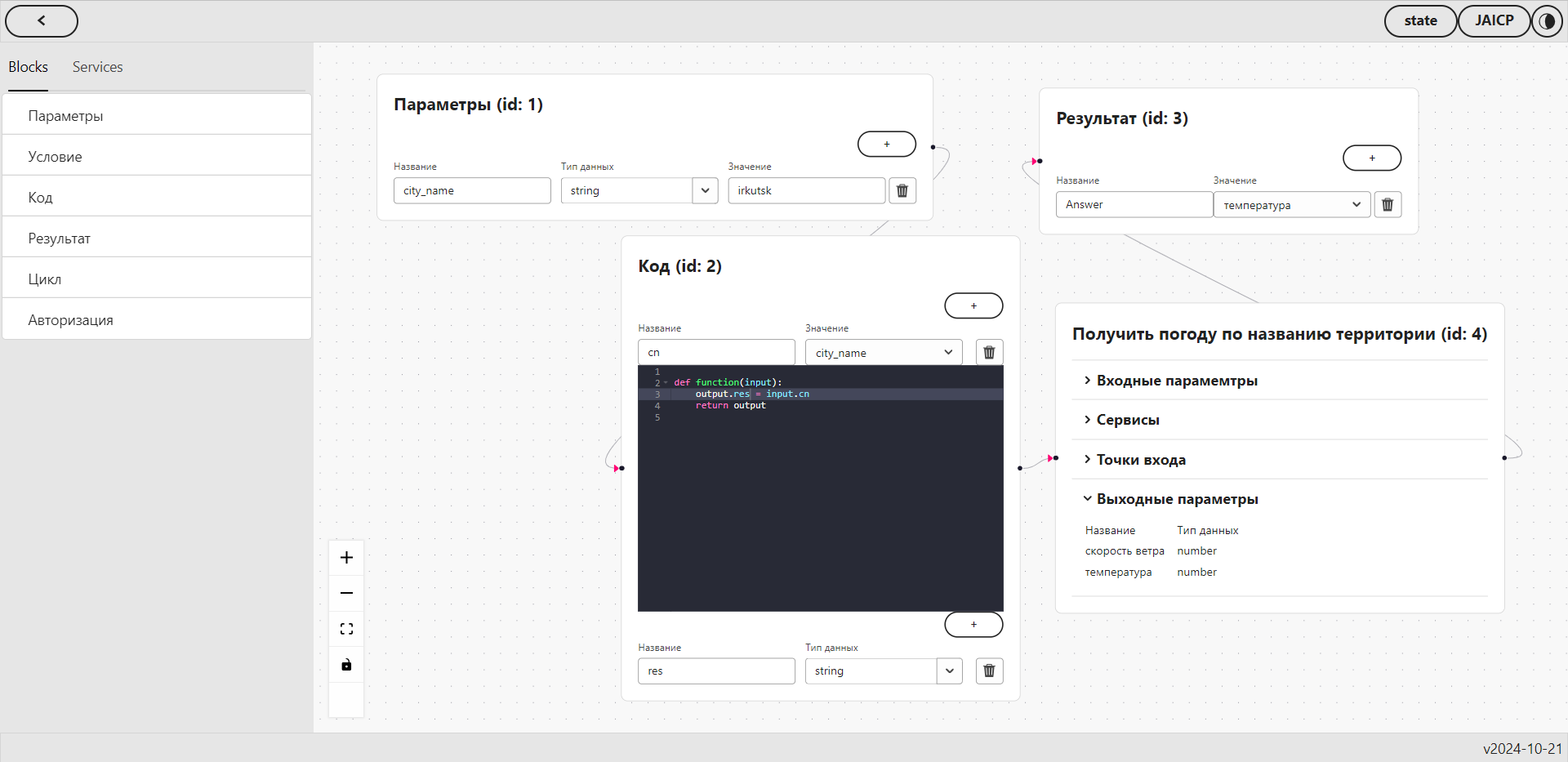


Рисунок 3.1 – Главное окно редактора

На рисунке 3.2 показан пример панели, отображающей код на языке JAICP DSL. Блок, предназначенный для решения поставленной задачи, генерирует код, который может быть интегрирован и использован на платформе JAICP.

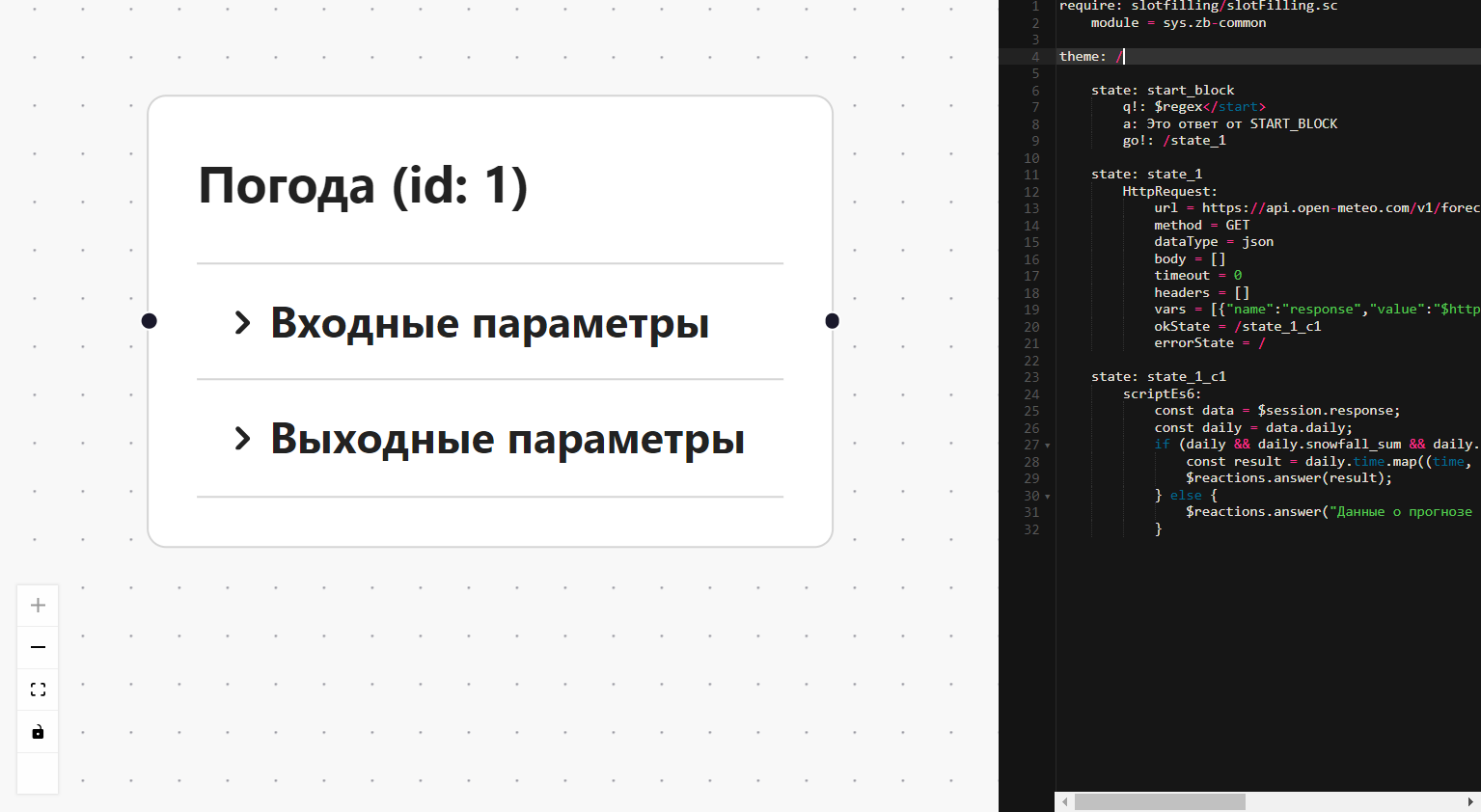


Рисунок 3.2 – Панель со сгенерированным кодом

На рисунке 3.3 показан код, который сгенерирован редактором и может быть использован на платформе.

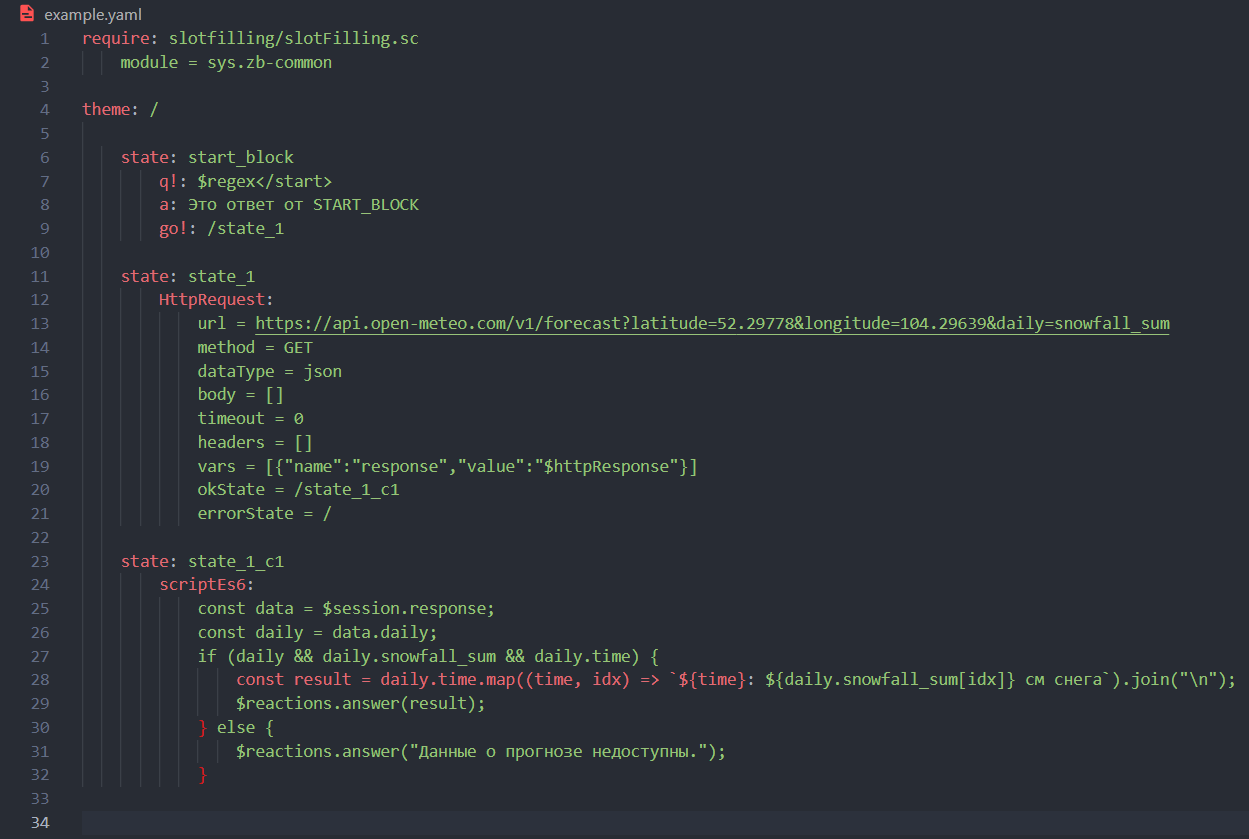


Рисунок 3.3 – Увеличенный пример сгенерированного кода

На рисунке 3.4 показаны доступные на данный момент простые блоки.

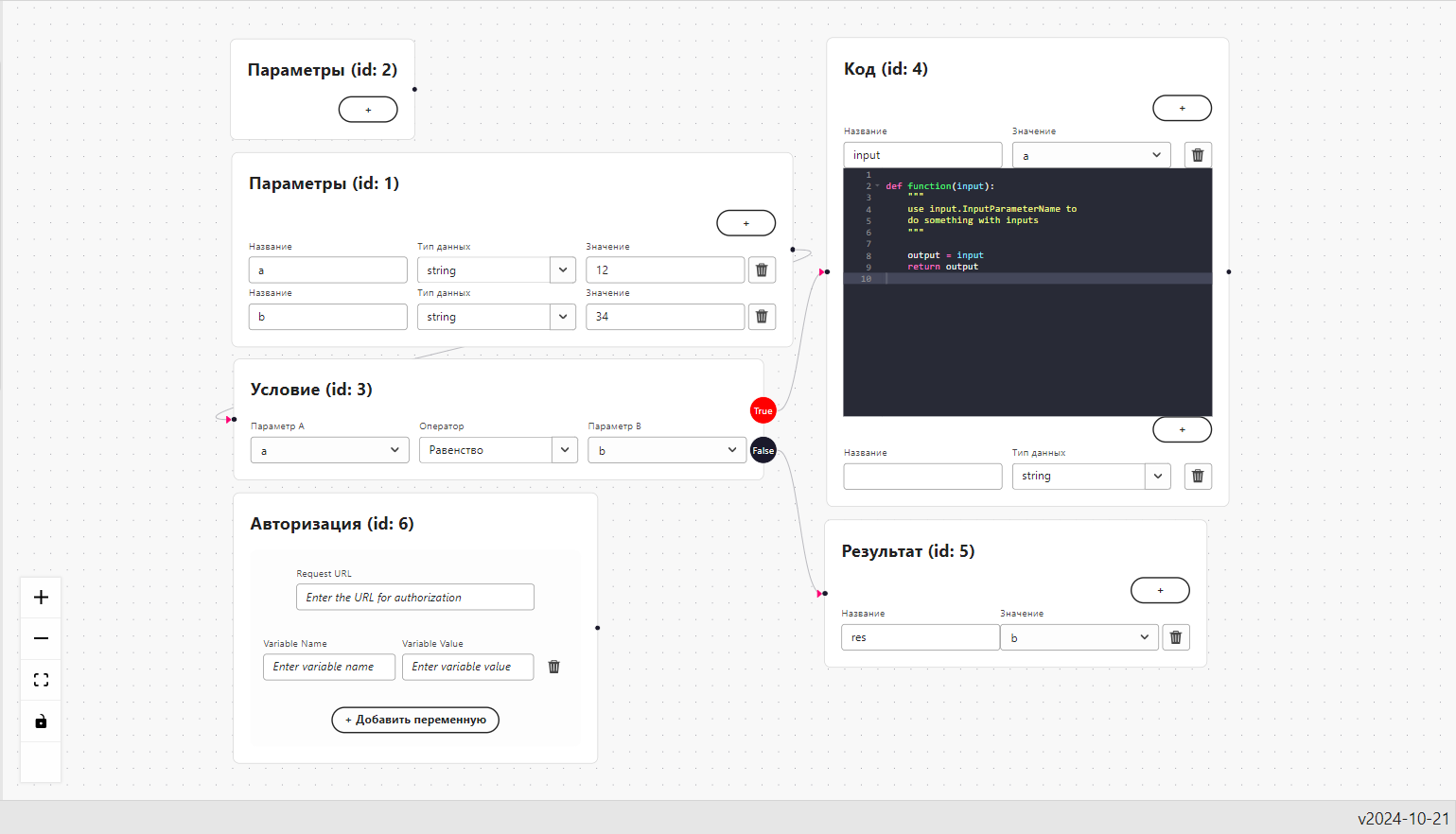


Рисунок 3.4 – Примеры типовых блоков

На рисунке 3.5 показана группа блоков, содержащая функцию и блок для выбора намерений, активирующих функцию. Эти намерения используются генератором, чтобы создавать условия, при которых виртуальный помощник выполнит функцию. Это позволит настроить работу с интерфейсом ввода информации пользователем помощника.

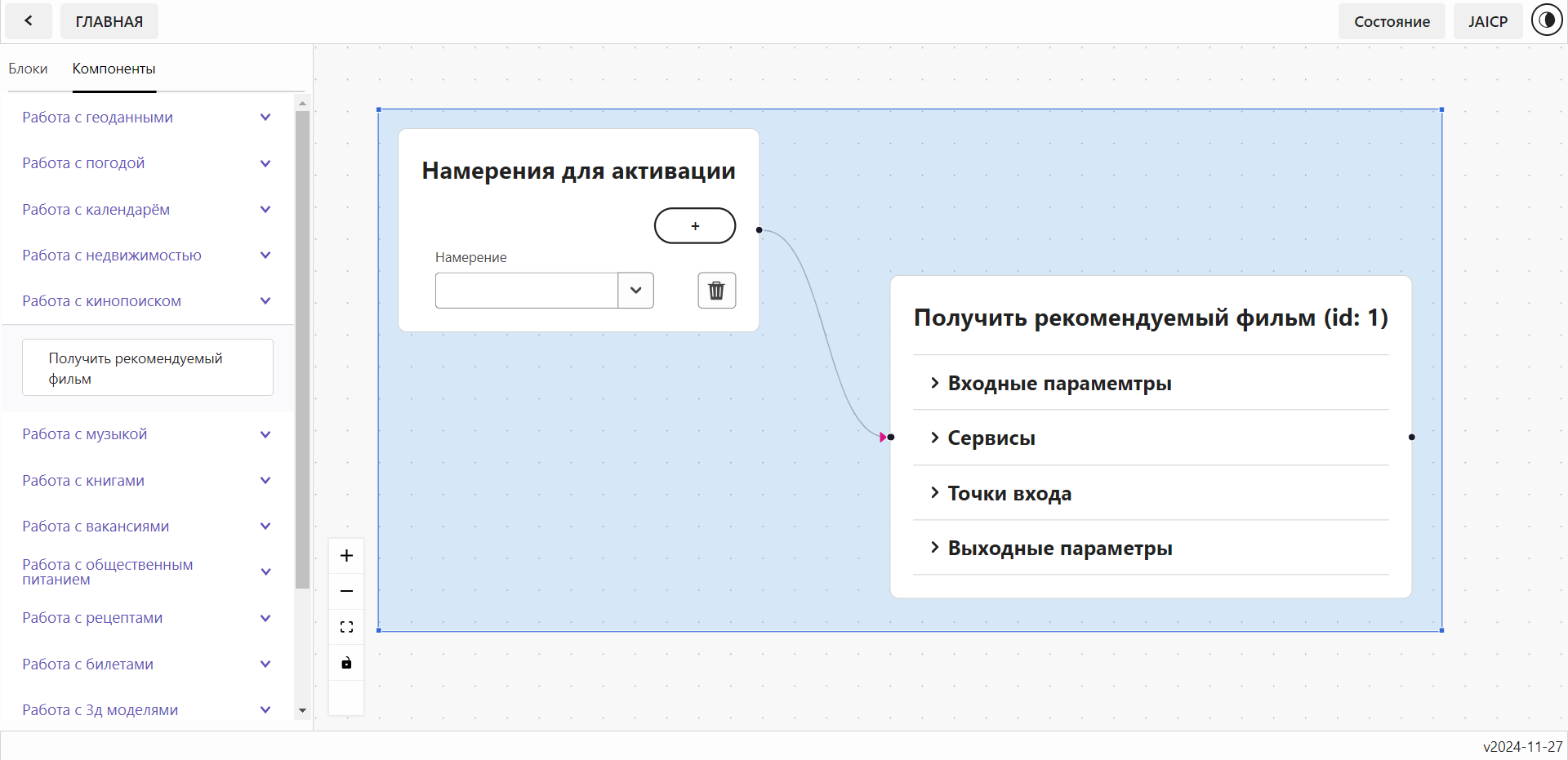


Рисунок 3.5 – Группа блоков

# Заключение

В результате выполнения курсовой работы было проведено исследование предметной области применения виртуальных ассистентов, рассмотрены аналогичные системы и были определены задачи и цель. Определены такие преимущества разрабатываемого редактора сценариев как упрощение работы с моделями данных и автоматизация большинства рутинных задач при создании схем работы виртуальных помощников.

Для редактора были реализованы как типовые блоки для управления логикой работы, так и блоки для решения классов задач.

Веб-приложение редактора обладает интуитивно понятным интерфейсом, доступным для разных групп пользователей.

Для обеспечения совместимости основные части системы взаимодействуют через API. Для удобства пользователей поддерживается экспорт в файлы JSON и DSL.

В результате создан редактор, который выполняет поставленные задачи и позволяет создавать сценарии для работы виртуальных ассистентов. Редактор позволяет генерировать код в спецификации языка DSL для использования на платформе JAICP.

# Список использованных источников­

1. Гутгарц Р.Д Проектирование автоматизированных систем обработки информации и управления: учебное пособие для академического бакалавриата. – М.: Издательство Юрайт, 2019.
2. Проектирование АСОИУ [Электронный ресурс]. Режим доступа: https://el.istu.edu/course/view.php?id=2770 (дата обращения: 07.11.2024).
3. О JAICP [Электронный ресурс]. Режим доступа: https://help.cloud.just-ai.com/jaicp/ (дата обращения: 15.11.2024).
4. Difference between Dialogflow CX vs Dialogflow ES [Электронный ресурс]. Режим доступа: https://dev.to/dhruv\_rajkotia/difference-between-dialogflow-cx-vs-dialogflow-es-3n1k (дата обращения: 15.11.2024).
5. Microsoft bot framework [Электронный ресурс]. Режим доступа: https://dev.botframework.com/#quick-starts (дата обращения: 07.11.2024).
6. Обзор Яндекс.Диалоги [Электронный ресурс]. Режим доступа: https://yandex.ru/dev/dialogs/alice/doc/ru/about (дата обращения: 07.11.2024).