



Руководство пользователя

GRAVITONUM PLATFORM

Содержание

О платформе.....	5
Глоссарий	6
Преимущества платформы	9
Традиционные преимущества low-code платформ	9
Преимущества low-code платформы Gravitonum platform	9
Область применения Gravitonum platform	9
MDM решения	9
Личный кабинет.....	10
Продуктовый каталог/ Тарификатор	11
Кадровый Электронный Документооборот (КЭДО).....	12
Архитектура приложения	12
Модель данных & API.....	13
Основные понятия	14
Сущности.....	14
Атрибуты сущности	14
Типы данных	14
Связи между сущностями.....	15
Один-к-одному.....	15
Один-ко-многим	16
Многие-ко-многим	16
Настройка модели данных.....	17
Сущности.....	17
Создание сущности	17
Настройка сущности	17
Удаление сущности	17
Атрибуты сущности	18
Создание атрибута.....	18
Настройка атрибута.....	20
Удаление атрибута.....	21
Экспорт	21
REST / GraphQL API.....	21
Rest API.....	21
GraphQL.....	23
Валидация.....	23
Валидация данных	23
Правила валидации.....	24
Настройка валидации на экранной форме	24
Пример правил валидации данных	25
Экранные формы (Компоненты).....	26

Создание и редактирование бизнес-компонента.....	26
Работа с дизайнером интерфейсов.....	27
Навигация (Блок 1).....	27
Канвас (Блок 2).....	28
Параметры виджета (Блок 3).....	32
Описание внешних переменных.....	33
Добавление внешней переменной.....	33
Создание значений по умолчанию.....	34
Работа с формулами в свойствах виджетов.....	34
Объединение нескольких полей в одно.....	34
Изменение регистра.....	35
Перевод на другую страницу.....	35
Интервал.....	35
Бизнес-процессы.....	36
Элементы BPMN нотации.....	36
События.....	37
Задачи.....	37
Шлюзы.....	38
Поток управления.....	38
Создание бизнес-процесса.....	39
Создание DMN моделей.....	43
Decision Table.....	43
Literal Expression.....	44
Деплой диаграммы на сервер.....	45
Как вызвать DMN через REST.....	45
Настройка ServiceTask.....	46
Настройка UserTask.....	47
Настройка бизнес-задачи.....	47
Json трансформация.....	49
Как перейти в трансформации?.....	49
Как работать с json трансформацией?.....	50
Добавление трансформации.....	50
Добавление параметров трансформации.....	52
Правила Json трансформации.....	52
Бизнес-приложение.....	54
Как запустить бизнес-приложение?.....	54
Как настроить бизнес-приложение?.....	54
Конфигурация меню.....	55
Шаблоны / Отчеты.....	66
Безопасность.....	68

Пользователи.....	68
Роли.....	68
Правила.....	69
Интеграция	70
Сервисы уведомлений	70
Импорт данных из Excel	70
Что требуется для импорта данных?	70
Настройка запроса	71
Рекомендации и параметры для конвертации	75
Хранилище ресурсов	76
Загрузка файлов	76
Стили интерфейса.....	80

О платформе

Платформа **Gravitonum platform** для разработки бизнес-приложений — это система нового поколения для автоматизации бизнес-процессов как в средних и крупных компаниях различных отраслей, так и в малом бизнесе.

Ключевым фактором, влияющим в наши дни практически на все отрасли, является цифровая трансформация. По мере того, как на рынке появляются различные цифровые технологии, растет и зависимость компаний от них. Сегодня все ведущие компании полагаются на цифровые решения, чтобы оставаться конкурентоспособными, упрощая свои процессы и сокращая время и ресурсы. Благодаря цифровизации, компании разрабатывают приложения для предоставления своим клиентам лучших решений и самое главное разрабатывают их быстро, как минимум, не отставая от конкурентов.

Все это приводит к растущему спросу на платформы разработки с низким уровнем кода как в России, так и во всем мире.

Концепция low-code позволяет существенно сократить время разработки бизнес-приложения, значительно сократив интервал времени от сбора бизнес-требований до работающего решения. Low-code платформа предназначена для «гражданских разработчиков» или тех, кто имеет небольшой опыт разработки или вообще не имеет его. Идея состоит в том, чтобы пользователи платформы сами создавали приложения, основанные на рабочих процессах. В рамках этого подхода, Аналитик как пользователь платформы не проводит сбор требований и не пишет детальное ТЗ для разработчиков, а потом не ждет, пока разработчики это ТЗ реализуют. С нашей платформой, компании получают высоко-функциональный инструмент, в котором «гражданские разработчики» могут сами, в визуальном конструкторе создавать работающие приложения.

Важно понимать, что low-code не обесценивает важность разработчиков программного обеспечения. Подход избавляет от необходимости писать и переписывать монотонный код, общий для всех программных продуктов. Вместо этого, «гражданские разработчики» работают с библиотекой настраиваемых компонентов, используя визуальный интерфейс для быстрого создания программного обеспечения. Отсутствие кодирования дает пользователю платформы сверх способности, позволяя за одну-две недели сделать то, что в при традиционном подходе заняло бы месяцы разработки большой командой. Это может иметь огромное влияние на прибыль компании. Быстрое достижение результатов за небольшую стоимость определенно даст преимущество компании на рынке.

Gravitonum platform состоит из следующих модулей:

1. **Data Designer** - конструктор структуры данных;
2. **UI Form Builder** - конструктор визуальных форм;
3. **BPMN/DMN Engine** - встроенный “движок” бизнес-процессов;
4. **Database** - реляционная база данных;
5. **Шаблоны** – модуль настройки шаблонов для уведомлений и отчетов;
6. **Security & Identity manager** - компонент информационной безопасности.

Свежую версию документации вы можете найти по ссылке
<https://doc.gravitonum.com/ru/latest/docs/intro>

Глоссарий

Термин	Определение
Low-code platform Цифровая трансформация	Платформа с минимальным уровнем кодирования процесс полной замены ручных, традиционных и устаревших способов ведения бизнеса новейшими цифровыми альтернативами
гражданский разработчик (Citizen Developer)	означает любого начинающего программиста или не-разработчика, на которого возложена ответственность за конфигурирование пользовательских приложений. Этот человек знаком с программированием, но не проходил профильного обучения
ТЗ	Техническое задание (ТЗ, техзадание) — документ или несколько документов, определяющих цель, структуру, свойства и методы какого-либо проекта, и исключающие двусмысленное толкование различными исполнителями.
Бизнес-процесс	совокупность взаимосвязанных мероприятий или работ, направленных на создание определённого продукта или услуги для потребителей. Управленческая концепция BPM (от англ. business process management) рассматривает бизнес-процессы как важные ресурсы предприятия, и предполагает управление ими как одну из ключевых организационных систем
Java	строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems. Разработка ведётся сообществом, организованным через Java Community Process; язык и основные реализующие его технологии распространяются по лицензии GPL
Angular	открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компаний. Angular — полностью переписанный фреймворк от той же команды, которая написала AngularJS
Quarkus	новый и очень быстрый фреймворк Java от компании RedHat, базируется на GraalVM и OpenJDK HotSpot и предназначен для Kubernetes. Стек Quarkus включает в себя: JPA/Hibernate, JAX-RS/RESTEasy, Eclipse Vert.x, Netty, Apache Camel, Kafka, Prometheus и другие.
OpenJDK	проект по созданию полностью совместимого Java Development Kit, состоящего исключительно из свободного и открытого исходного кода.
GraalVM	это виртуальная машина Java и JDK, основанная на HotSpot/OpenJDK и написанная на Java. GraalVM поддерживает разные языки программирования и модели выполнения, такие как JIT-компиляция и AOT-компиляция. Первая стабильная версия, 19.0, была выпущена в мае 2019-ого года.

Термин	Определение
GraphQL	это язык запросов данных и манипулирования ими с открытым исходным кодом для API, а также среда выполнения для выполнения запросов с существующими данными.
Rest API	архитектурный стиль взаимодействия компонентов распределённого приложения в сети. Другими словами, REST — это набор правил того, как программисту организовать написание кода серверного приложения, чтобы все системы легко обменивались данными и приложение можно было масштабировать
Kubernetes	открытое программное обеспечение для оркестровки контейнеризированных приложений — автоматизации их развёртывания, масштабирования и координации в условиях кластера. Поддерживает основные технологии контейнеризации, включая Docker, rkt, также возможна поддержка технологий аппаратной виртуализации
Docker	программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений.
IDE	интегрированная среда разработки (IDE) — это программное приложение, предоставляющее программистам комплексные средства для разработки программного обеспечения. IDE обычно состоит как минимум из редактора исходного кода, средств автоматизации сборки и отладчика
single sign-on	технология единого входа — технология, при использовании которой пользователь переходит из одного раздела портала в другой, либо из одной системы в другую, не связанную с первой системой, без повторной аутентификации.
MDM (Master Data Management)	управление основными данными (англ. Master Data Management, MDM) — высокотехнологичная дисциплина на стыке бизнес-управления и ИТ, призванная обеспечить единообразие, точность, ответственность, семантическую согласованность и подотчетность распространяемых или открываемых для совместного доступа официальных основных данных, имеющихся в активе организации.
json	текстовый формат обмена данными, основанный на JavaScript. Как и многие другие текстовые форматы, JSON легко читается людьми. Формат JSON был разработан Дугласом Крокфордом.
валидация	это процесс проверки данных различных типов по критериям корректности и полезности для конкретного применения.
миграция	это процесс перемещения больших объемов данных из одного места в другое.
Message Queue	очередь сообщений — в информатике — программно-инженерный компонент, используемый для межпроцессного или межпоточкового взаимодействия внутри одного процесса. Для обмена сообщениями используется очередь
Business-to-Employee	система взаимодействия предприятия с работниками, которая использует внутрикорпоративную сеть и позволяет компаниям предоставлять продукты и/или услуги своим сотрудникам. Как

Термин	Определение
.odt	<p>правило, компании используют сети В2Е для автоматизации корпоративных процессов, связанных с сотрудниками.</p> <p>OpenDocument Format, ODF — открытый формат файлов документов для хранения и обмена редактируемыми офисными документами, в том числе текстовыми документами, электронными таблицами, рисунками, базами данных, презентациями.</p>
stateless	<p>это принцип проектирования, который применяется в рамках парадигмы проектирования, ориентированной на службы, для разработки масштабируемых служб путем отделения их от данных о состоянии, когда это возможно.</p>
Вертикальное масштабирование	<p>наращивание или снижение вычислительной мощности или ресурсов баз данных по мере необходимости</p>
Горизонтальное масштабирование	<p>разбиение системы на более мелкие структурные компоненты и разнесение их по отдельным физическим машинам (или их группам), и (или) увеличение количества серверов, параллельно выполняющих одну и ту же функцию.</p>

Преимущества платформы

Традиционные преимущества low-code платформ

1. Платформа сокращает затраты времени на разработку и внедрение нового продукта, а также на поиск и найм команды разработки.
2. Платформа автоматически генерирует код, общий для большинства программных продуктов, поэтому разработчики тратят меньше времени на подготовку проекта.
3. Не нужны в большом количестве дорогостоящие разработчики фронт-офиса и бэк-офиса.
4. Можно быстро вносить изменения в разработанные решения.

Преимущества low-code платформы Gravitonum platform

1. Разработанное бизнес-приложение является отчуждаемым, может работать и дорабатываться без использования платформы. Платформа выступает в роли онлайн среды разработки, которая генерирует Java и Angular код, работающий самостоятельно;
2. Код запускается в супербыстром и современном фреймворке Quarkus под Open JDK или GraalVM, что позволяет мгновенно ре-стартовать приложения, имеющие при этом небольшой размер;
3. Автоматически генерируется GraphQL и REST API;
4. Контейнеризация позволяет развернуть решение на таких платформах как OpenShift (OKD), Kubernetes, Docker Swarm;
5. Механизм обратной совместимости позволяет редактировать код приложения в стандартных IDE и видеть изменения в платформе.

Область применения Gravitonum platform

Результатом работы Gravitonum platform является самостоятельное приложение в концепции micro-app (микроприложение).

Микроприложение — это специализированное приложение, предназначенное для выполнения конкретно задачи с единственной целью — сделать это хорошо.

Микроприложения могут быть объединены единым порталом с механизмом single sign-on, позволяющим конечным пользователям работать с несколькими приложениями удобным для них способом.

Наиболее эффективно платформа может быть использована при разработке приложений, в которых есть сложные бизнес-процессы, как с участием пользователей (User Task), так и вызовами внешних систем (Service Task), сложные структуры данных, импорт из других систем с валидацией данных и разнообразные экраны формы.

Рассмотрим варианты микроприложений, для разработки которых может быть использована **Gravitonum platform**

MDM решения

MDM-системы — это решения для управления мастер данными. Их главная цель — обеспечить единство представления массивов данных во всех информационных

системах. Кроме того, такой тип решений позволяет решить проблемы несоответствия, дублирования и несопоставимости данных.

Мастер-данные («основные данные» или «нормативно-справочная информация») — это данные, записывающие справочную информацию, то есть значения, которые могут использоваться для указания, к чему какие данные относятся. Самый простой пример применения мастер-данных – разного рода справочники или классификаторы.

Платформа позволяет быстро создавать решения класса MDM (Master Data Management), за счет следующих факторов:

1. визуальный конструктор модели данных позволяет быстро создать структуру справочников
2. механизмы импорта данных из excel, cvs, json файлов позволяет мигрировать данные из других систем
3. механизм управления событиями позволяет отслеживать операции создания/изменения/удаления записей и отправлять изменения в системы потребители
4. встроенный механизм интеграции с решениями класса Message Queue позволяет рассылать данные по изменениям централизованным способом
5. GraphQL и REST API позволяют легко предоставить данные внешним системам потребителям, не умеющим работать с решениями класса Message Queue
6. механизм валидации позволяет визуально задать необходимые проверки для контроля импорта данных в справочники
7. автоматическая генерация форм позволяет легко создать интерфейсы для ручного редактирования справочников
8. механизм управления бизнес-процессами позволяет создать процессы многоступенчатого согласования данных в справочниках

Личный кабинет

Личный кабинет - это решение класса B2E (Business-to-Employee), которое автоматизирует рабочее место сотрудника компании.

Личный кабинет определяет права и возможности сотрудника.

Платформа позволяет быстро создавать решения класса Личный кабинет, используя такие преимущества платформы как:

1. визуальный конструктор модели данных позволяет быстро создать структуру данных для хранения информации о пользователях
2. механизмы импорта данных из excel, cvs, json файлов позволяет импортировать данные из других систем, таких как 1С и т.д.
3. высоко-функциональный дизайнер интерфейсов позволяет создавать экраны любой сложности, за счет использования широкой палитры готовых компонент и возможности стилизации под требования заказчика
4. встроенный механизм интеграции с решениями класса Message Queue позволяет рассылать данные в другие системы централизованным способом

5. GraphQL и REST API позволяют легко предоставить данные внешним системам потребителям, не умеющим работать с решениями класса Message Queue
6. механизм управления бизнес-процессами позволяет создать процессы многоступенчатого согласования документов
7. платформа позволяет отследить задачи назначенные на пользователя и уведомить их как внутри решения так и по внешним каналам связи
8. механизм шаблонов позволяет создать шаблоны уведомлений и отчеты для выгрузки в формате .odt
9. механизм уведомлений позволяет сделать рассылку сообщений через email, sms, telegram.
10. готовый адаптер позволяет подключить пользователей через ЕСИА
11. поддержка ролевой модели доступа, вплоть до атрибута, как в back-office так и во front-office

Продуктовый каталог/ Тарификатор

Продуктовый каталог - это современный инструмент управления информацией о продуктах предприятия. По мере развития предприятия продуктовые предложения объединяются в продуктовые линейки, а затем формируется общий каталог предложений банка - каталог продуктов.

Тарификатор – это система, предназначенная для online или offline расчета комиссий для полученных транзакций, на основании данных тарифов (продуктовый каталог)

Платформа позволяет быстро создавать решения класса Продуктовый каталог и Тарификатор, используя механизмы:

1. визуальный конструктор модели данных для создания структуры данных хранения информации о продуктах
2. высоко-функциональный дизайнер интерфейсов позволяет создавать экраны любой сложности для участников процесса, за счет использования широкой палитры готовых компонент и возможности стилизации под требования заказчика
3. поддерживается иерархическое ведение справочников тарифов с механизмом наследования
4. ведется учет исторических данных для отслеживания таких параметров продуктов как количество бесплатных транзакций, комиссий в зависимости от суммы, и т.п.
5. механизмы импорта данных, позволяющих импортировать данные из других систем
6. механизм валидации позволяет визуально задать необходимые проверки для контроля импорта данных в справочники
7. поддержка DMN (Decision Management Notation) позволяет создать необходимые проверки поступающих из вне данных в онлайн режиме

8. автоматическая генерация форм позволяет легко создать интерфейсы для ручного редактирования справочников
9. механизм управления бизнес-процессами позволяет создать процессы многоступенчатого согласования данных в справочниках
10. механизм шаблонов позволяет создать шаблоны уведомлений и отчеты для выгрузки в формате .odt
11. механизм уведомлений позволяет сделать рассылку сообщений через email, sms, telegram.

Кадровый Электронный Документооборот (КЭДО)

КЭДО - это класс систем для автоматизации электронного документооборота с работниками (договоры, больничные, отпуска, справки, итд).

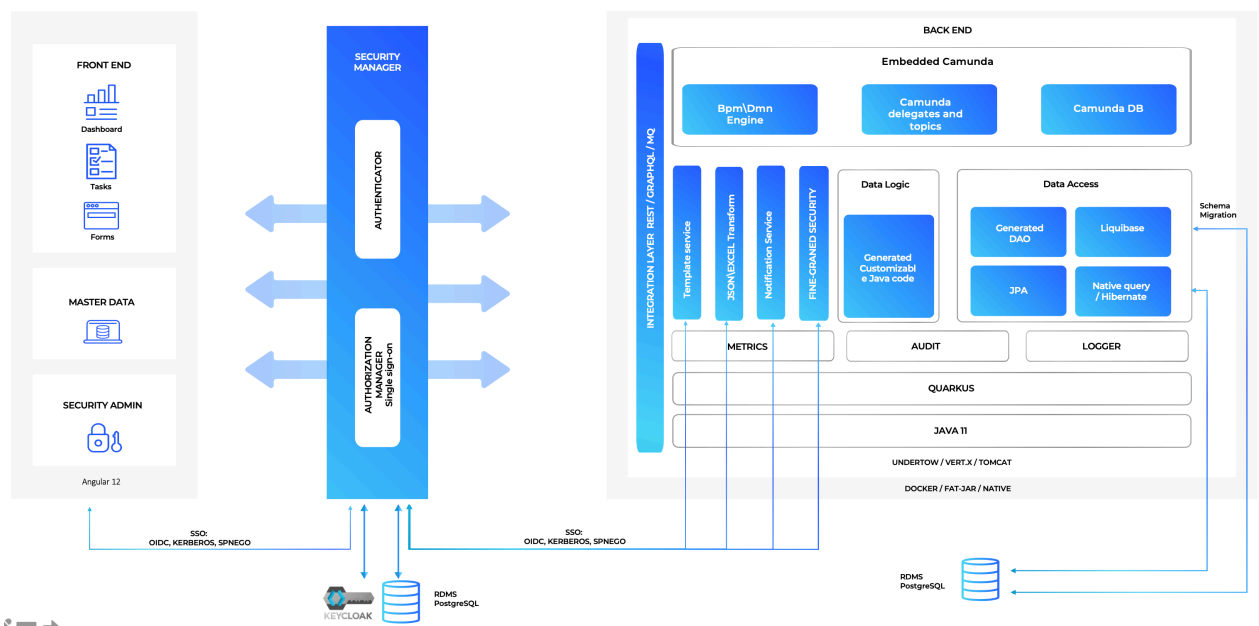
Платформа позволяет быстро создавать решения класса Электронный Документооборот для внутренних нужд, используя механизмы:

1. визуальный конструктор модели данных для создания структуры данных хранения информации о согласуемых документах
2. высоко-функциональный дизайнер интерфейсов позволяет создавать экраны любой сложности для участников процесса, за счет использования широкой палитры готовых компонент и возможности стилизации под требования заказчика
3. поддержка внешних провайдеров УНЭП/УКЭП
4. механизмы импорта данных, позволяющих импортировать данные из других систем, таких как 1С
5. механизм управления бизнес-процессами позволяет создать процессы многоступенчатого согласования документов
6. механизм шаблонов позволяет создать шаблоны уведомлений и отчеты для выгрузки в формате .odt
7. механизм уведомлений позволяет сделать рассылку сообщений через email, sms, telegram.

Архитектура приложения

Архитектура платформы обеспечивает высокую масштабируемость решений и поддерживает оба типа масштабирования: горизонтальное и вертикальное, с упором на горизонтальное масштабирование. Все модули(сервисы) являются stateless, экземпляры работают в легковесных контейнерах Docker, без связи между модулями на уровне базы данных.

Высокоуровневая архитектура платформы представлена на Рисунке ниже



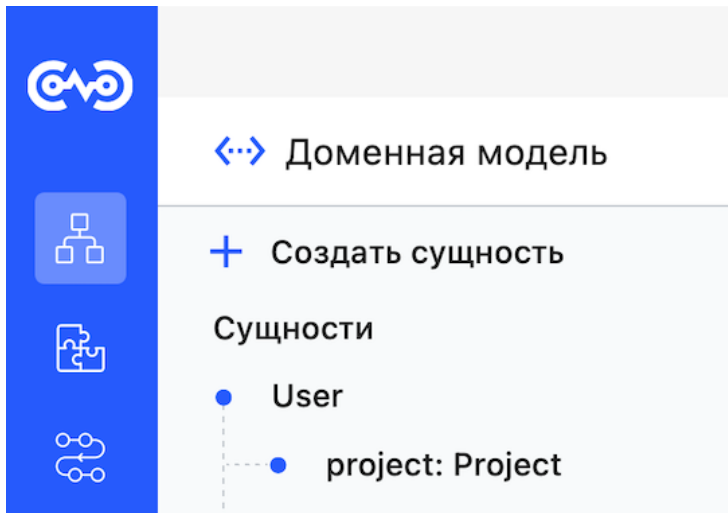
Модель данных & API

Модель данных - это структура схемы базы данных, в которой хранятся данные бизнес-приложения.

Дизайнер модели данных позволяет в онлайн режиме:

- Создавать/удалять/настраивать/переименовывать сущности (таблицы) и их атрибуты (столбцы таблицы).
- Настраивать связи между сущностями.
- Автоматически создавать REST/GraphQL API для работы с сущностями.
- Добавлять собственные SQL запросы, например, для графиков, и кастомные Java-методы.
- Создавать события жизненного цикла сущности (аналог триггеров в БД).
- Настраивать правила валидации всех сущности или части ее атрибутов.
- Автоматически создавать пользовательский интерфейс для сущности.

Дизайнер модели данных является первым элементом меню основного меню платформы, находящегося в левой части экрана.



Меню “Модель”

ОСНОВНЫЕ ПОНЯТИЯ

Сущности

Сущность – любой конкретный или абстрактный объект в рассматриваемой предметной области, который должен хранить свое состояние в БД.

В реляционной БД для каждой сущности создается своя таблица.

Каждая сущность имеет наименование, выраженное существительным в единственном числе на английском языке.

Примерами сущностей могут быть такие классы объектов как Supplier, Employee, SalesOrder.

Название сущности пишется с большой буквы в Camel-регистре (CamelCase).

Атрибуты сущности

Атрибут сущности - это именованная характеристика, являющаяся некоторым свойством сущности. В реляционной БД для каждого атрибута сущности создается столбец.

Наименование атрибута должно быть выражено существительным в единственном числе (возможно, с характеризующими прилагательными).

Примерами атрибутов сущности Employee могут быть такие атрибуты как personalNumber, lastName, firstName, middleName, position, monthlySalary и т.п.

Название атрибута пишется с маленькой буквы в нижнем Camel-регистре (lowerCamelCase).

Атрибут сущности всегда имеет *тип данных*.

Типы данных

Тип данных (тип) — множество значений и операций над этими значениями.

К основным типам данных относятся:

— Логический тип - может иметь лишь одно из двух состояний — true или false –

- Целочисленные типы - содержат в себе значения, интерпретируемые как целые числа (знаковые и без знаковые). Например, 5, 7371, 100000.
- Числа с плавающей запятой - используются для представления вещественных (не обязательно целых) чисел. 3.14, 122.5.
- Строковые типы, например, “Иванов Иван Иванович”, “Отдел качества”.

Общую информацию о типах данных можно уточнить на [Wikipedia](#).

Типы данных СУБД Postgres, которая поставляется по умолчанию, находятся [здесь](#).

Про типы данных Java можно почитать [здесь](#).

Связи между сущностями

Связь - некоторая ассоциация между двумя сущностями. Одна сущность может быть связана с другой сущностью или сама с собою.

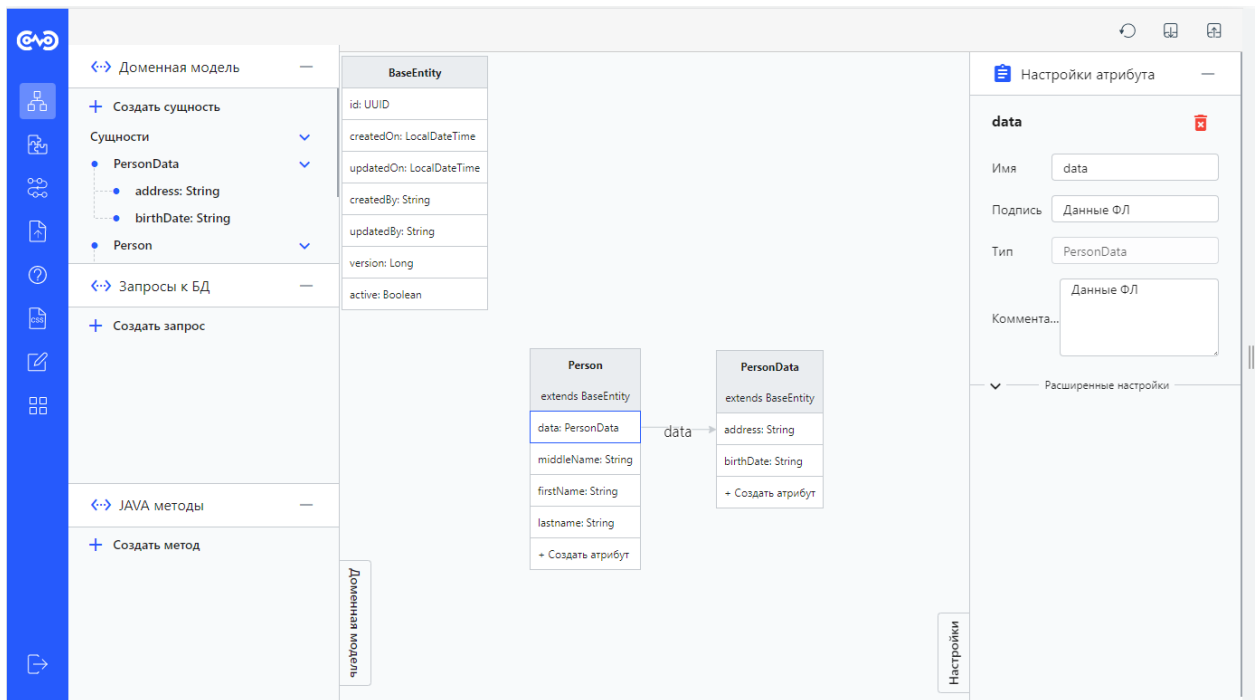
Связи позволяют по одной сущности находить другие сущности, связанные с нею. Например, связи между сущностями могут выражаться следующими фразами - “Сотрудник может иметь несколько Детей”, “каждый Сотрудник обязан числиться ровно в одном Отделе”.

Выделяют следующие типы связей:

- Один-к-одному
- Один-ко-многим
- Многие-ко-многим

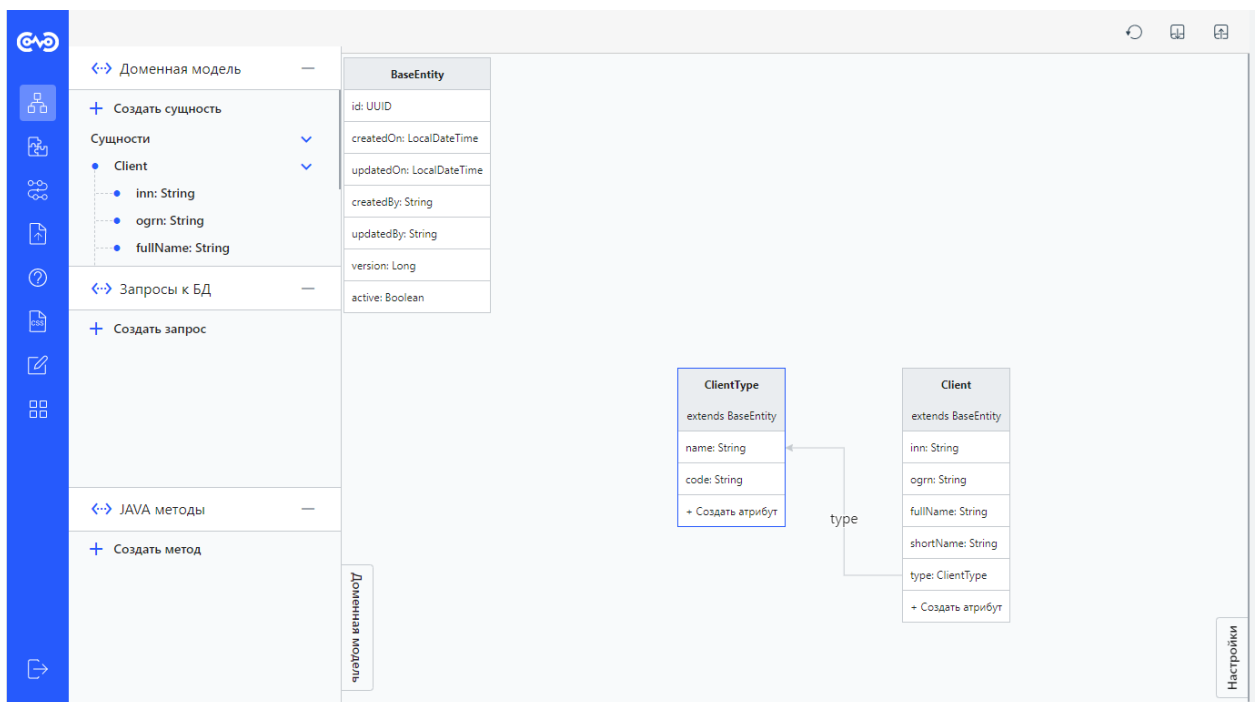
Один-к-одному

Связь типа *один-к-одному* означает, что один экземпляр первой сущности (левой) связан с одним экземпляром второй сущности (правой). Связь один-к-одному чаще всего свидетельствует о том, что на самом деле мы имеем всего одну сущность, по какой-то причине (безопасность, производительность) разделенную на две.



Один-ко-многим

Связь типа *один-ко-многим* означает, что один экземпляр первой сущности (левой) связан с несколькими экземплярами второй сущности (правой). Это наиболее часто используемый тип связи. Левая сущность (со стороны “один”) называется родительской, правая (со стороны “много”) - дочерней.



Многие-ко-многим

Связь типа *многие-ко-многим* означает, что каждый экземпляр первой сущности может быть связан с несколькими экземплярами второй сущности, и каждый экземпляр второй сущности может быть связан с несколькими экземплярами первой сущности. Тип связи много-ко-многим является логическим типом связи. В реальной модели данных этот тип связи представляется двумя связями типа *один-ко-многим* путем создания промежуточной сущности.

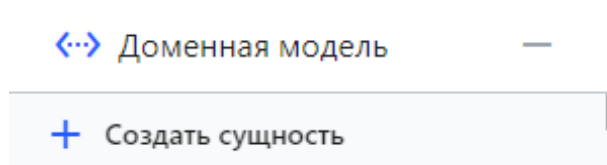
Настройка модели данных

Сущности

Общая информация о сущности приведена на странице [Основные понятия](#).

Создание сущности

Для того, чтобы создать новую сущность, необходимо выбрать пункт “Создать сущность с левой стороны интерфейса.



Создать сущность

в открывшемся окне необходимо заполнить следующие поля: - Имя - Подпись - Комментарий

Поле Имя определяет имя сущности, выраженное существительным в единственном числе на английском языке.

Примерами сущностей могут быть такие классы объектов как Supplier, Employee, SalesOrder.

Название сущности пишется с большой буквы в Camel-регистре (CamelCase).

В поле Подпись задается смысловое описание данной сущности. Описание может быть на как на русском, так и на английском языке.

Поле Комментарий определяет дополнительную информацию по сущности.

Настройка сущности

После создания сущности, есть возможность изменить часть параметров, таких как: - Имя - Подпись

После каждого изменения этих свойств, происходит регенерация, на основании новых данных.

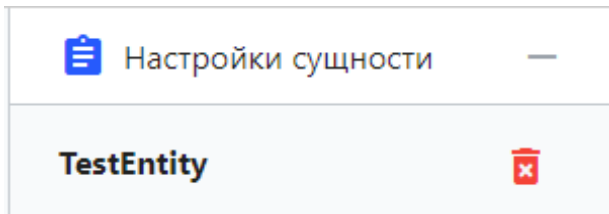
Дополнительно, можно настроить следующие свойства:

Справочник - свойство, определяющее поведение платформы, при операциях push/pull. Если флаг установлен, то при операциях push/pull происходит передача метаданных сущности вместе со значениями (записями). Это позволяет легко переносить справочные данных между средами, избегая двойного ввода.

Аудит - Установка данного флага, сигнализирует платформе, что по данной сущности необходимо вести учет (аудит) всех изменений.

Удаление сущности

Для удаления сущности, необходимо выбрать сущность, и справа в панели свойств нажать значок



Удалить

а после выбрать “ОК” в окне подтверждения

Удаление сущности: TestEntity

Вы действительно хотите удалить сущность: TestEntity?

ОК

Отменить

Подтверждение

Атрибуты сущности

Общая информация о атрибуте приведена на странице [Основные понятия](#).

Создание атрибута

Для того, чтобы создать новый атрибут сущности, необходимо выбрать сущность и нажать кнопку “+ Создать атрибут”

+ Создать атрибут

Создать атрибут

В открывшемся окне

TestEntity

Создать атрибут

Имя *

Тип *

String

Many to one

One to one

Подпись

Комментарий

Добавить обратную связь

Создать

Отменить

Создание атрибута

необходимо заполнить следующие поля:

- **Наименование** - наименование атрибута должно быть выражено существительным в единственном числе (возможно, с характеризующими прилагательными).

Примерами атрибутов сущности Employee могут быть такие атрибуты как personalNumber, lastName, firstName, middleName, position, monthlySalary и т.п.

Название атрибута пишется с маленькой буквы в нижнем Camel-регистре (lowerCamelCase).

- **Тип** - к основным типам данных относятся:
 - Boolean - Логический тип, может иметь лишь одно из двух состояний — true или false
 - Byte\Short\Integer\Long - Целочисленные типы, содержат в себе значения, интерпретируемые как целые числа (знаковые и без знаковые). Например, 5, 7371, 100000.
 - Bigdecimal\Float\Double - Числа с плавающей запятой, используются для представления вещественных (не обязательно целых) чисел. 3.14, 122.5. - String
 - Строковые типы, например, “Иванов Иван Иванович”, “Отдел качества”. – LocalTime

- Типы для хранения значений времени и даты, как по отдельности, так и вместе.
- UUID - Уникальный идентификатор - Byte[] - массив значений типа Byte, например картинка в бинарном виде - JSONB - данные в формате JSON

Также, в качестве типа атрибута, можно указать наименование уже созданной ранее Сущности, в этом случае откроются дополнительные поля

Тип *

Branch

Many to one

One to one

Связь

Необходимо выбрать тип связи один-к-одному или многие-к-одному

Добавить обратную связь

Обратная связь

В случае выбранного чек бокса и связи многие-к-одному, на правой стороне средствами GraphQL формируется список, который позволяет посмотреть список всех записей на левой стороне, ссылающиеся на запись на правой.

В поле **Подпись** задается смысловое описание данного атрибута. Описание может быть на как на русском, так и на английском языке.

Поле **Комментарий** определяет дополнительную информацию по атрибуту.

Порядок отображения атрибутов внутри сущности можно менять, перетаскивая атрибуты мышкой на новую позицию внутри сущности. Данный порядок влияет на визуальное отображение и на порядок генерации атрибутов в формах автоматической генерации.

Настройка атрибута

После создания атрибута, есть возможность изменить часть параметров, таких как:

- Наименование
- Подпись
- Комментарий
- Тип, если он простой, не ссылка

После каждого изменения этих свойств, происходит регенерация, на основании новых данных.

Дополнительно, можно настроить следующие свойства:

Расширенные настройки

Уникальный - при установке данного флага, проверяется уникальность значений поля при создании\изменении записи, а также дополнительно генерируется дополнительный GraphQL метод [entity]GetBy[имя атрибута].

Обязательный - при установке данного флага проверяется наличие значения при создании\изменении записи, значение null недопустимо.

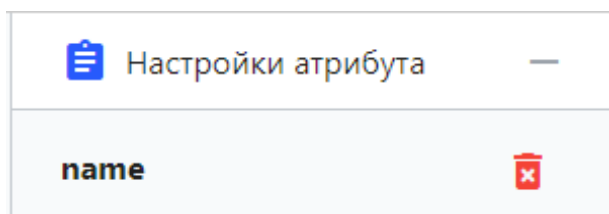
Тип в БД - позволяет поменять тип атрибута непосредственно в базе данных (Только для опытных пользователей)

Валидация

Настройка валидации детально описана в отдельном «Валидация»

Удаление атрибута

Для удаления сущности, необходимо выбрать сущность, и справа в панели свойств нажать значок



Удалить

а после выбрать “ОК” в окне подтверждения

Экспорт

Для импорта\экспорта модели данных есть 2 механизма:

1. Сохранение и загрузка данных модели путем интеграции с Git. Для работы с данным механизмом, нужно удостовериться, что включена опция платформы vcs. Уточните у администратора, доступна ли данная опция на вашем стенде. Кнопка push отправляет изменения в репозиторий. Кнопка pull предназначена для извлечения и загрузки содержимого из удаленного репозитория
2. Сохранение\Загрузка модели данных локально с помощью REST API
 - Экспорт GET `{{ generator }}/api/v1/data_model`
 - Импорт POST `{{ generator }}/api/v1/data_model`

REST / GraphQL API

Gravitonum platform позволяет осуществлять автоматическую генерацию REST API и GraphQL методов.

Rest API

Rest API - это способ взаимодействия сайтов и веб-приложений с сервером. Его также называют RESTful.

Термин состоит из двух аббревиатур, которые расшифровываются следующим образом. API (Application Programming Interface) — это код, который позволяет двум приложениям обмениваться данными с сервера.

На русском языке его принято называть программным интерфейсом приложения.

REST (Representational State Transfer) — это способ создания API с помощью протокола HTTP.

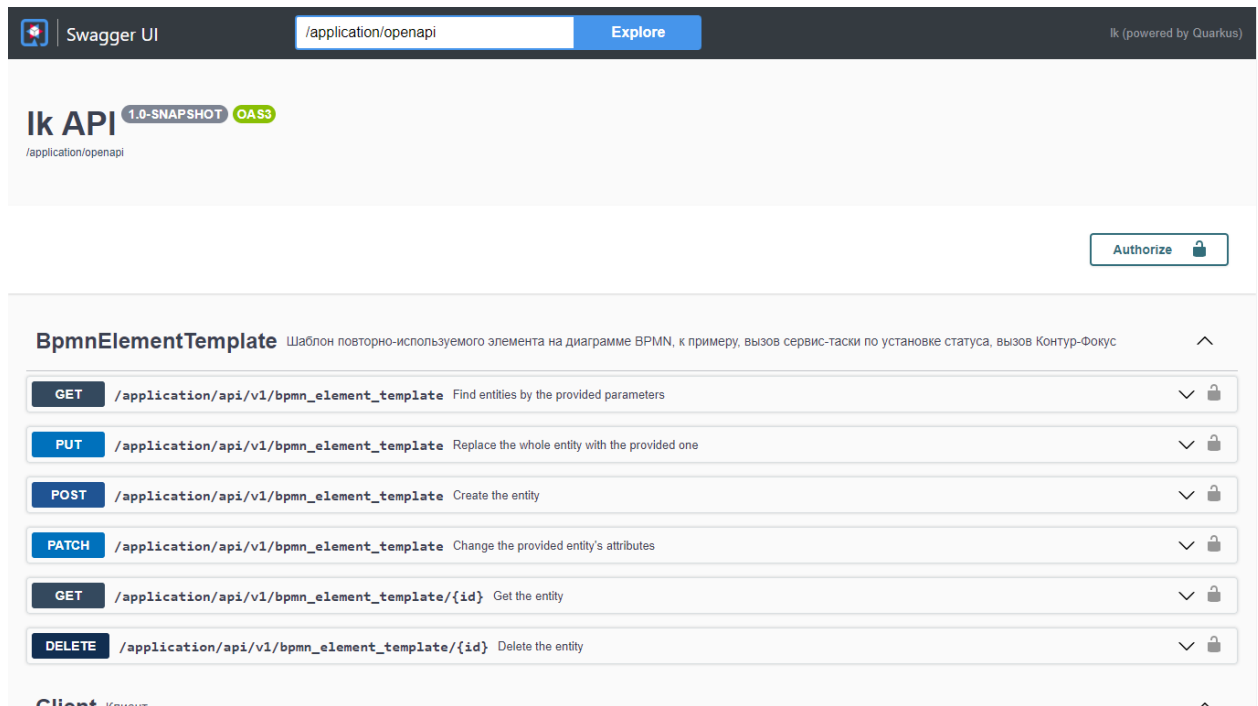
На русском его называют «передачей состояния представления».

В настоящее время это самый распространенный способ организации API. Он вытеснил ранее популярные способы SOAP и WSDL.

Платформа автоматически генерирует документацию (swagger) к REST API.

Swagger - это фреймворк для спецификации RESTful API. Его преимущество заключается в том, что он дает возможность не только интерактивно просматривать спецификацию, но и отправлять запросы – так называемый Swagger UI.

Доступ к swagger осуществляется по ссылке вида <host>/application/swagger-ui/ Пример документации приведен ниже.



Swagger

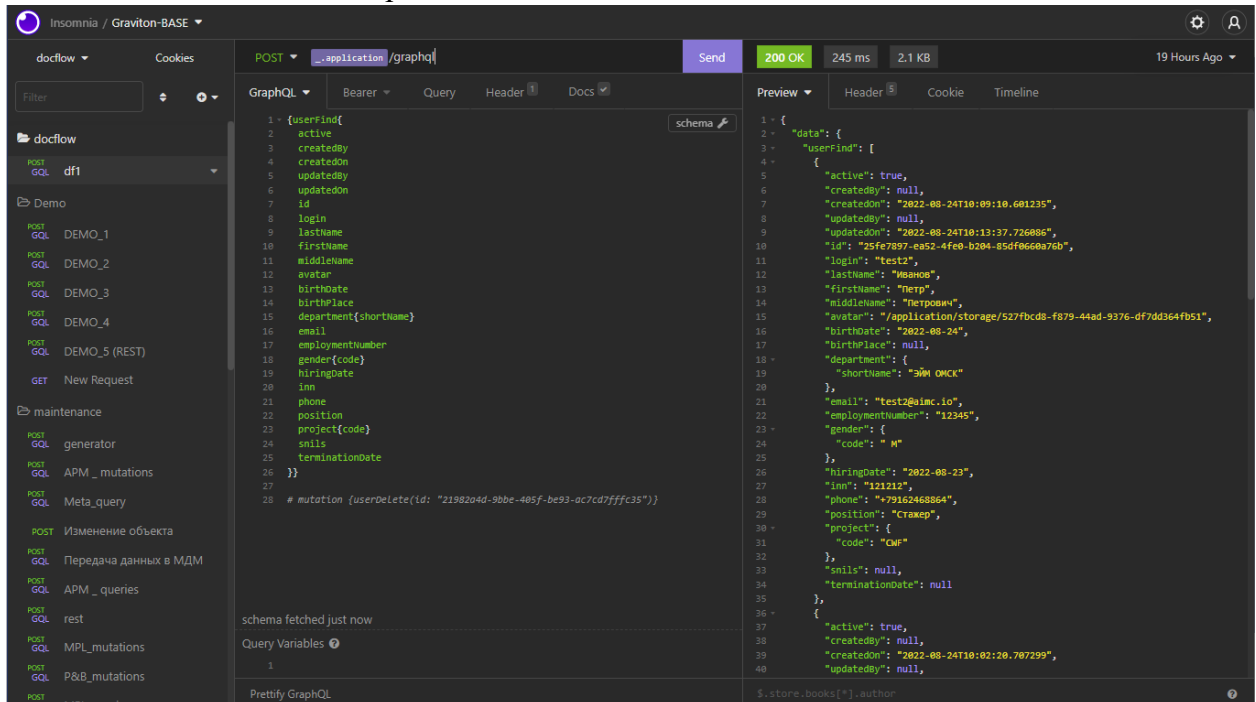
Платформа автоматически генерирует 6 методов для каждой сущности:

1. **GET** /application/api/v1/[Имя сущности] Поиск записей по параметрам (find)
2. **PUT** /application/api/v1/[Имя сущности] Полная замена записи (replace)
3. **POST** /application/api/v1/[Имя сущности] Создание новой записи (create)
4. **PATCH** /application/api/v1/[Имя сущности] Изменение атрибутов записи (update)
5. **GET** /application/api/v1/[Имя сущности]/{id} Получение данных записи по id (findByid)
6. **DELETE** /application/api/v1/[Имя сущности]/{id} Удаление записи (delete)

GraphQL

В двух словах, GraphQL это синтаксис, который описывает как запрашивать данные, и, в основном, используется клиентом для загрузки данных с сервера. GraphQL имеет три основные характеристики: - Позволяет клиенту точно указать, какие данные ему нужны - Облегчает агрегацию данных из нескольких источников. - Использует систему типов для описания данных.

Воспользоваться методами GraphQL можно с помощью настройки вызова непосредственно из экранных форм либо с помощью специальных внешних инструментов, таких как Insomnia или Postman, как представлено ниже.



Платформа автоматически генерирует 3 запроса (query) для каждой сущности:

1. [сущность]Count - Количество записей в сущности
2. [сущность]Find - Поиск записей по параметрам
3. [сущность]FindById- Получение данных записи по id

и 2 мутации (mutation) для каждой сущности:

1. [сущность]Save - Создание новой записи и ее изменение
2. [сущность]Delete - Удаление записи

Валидация

Валидация данных

Валидация данных (англ. Data validation) — это процесс проверки данных различных типов по критериям корректности и полезности для конкретного применения.

Наиболее очевидное использование валидации данных мы видим в валидации атрибутов сущности.

Валидация данных реализована на стороне back-office и может быть использована как в методах back-office так и на экранных формах. Вы можете создать сущность, создать

экранную форму и добавить правила проверки, которые будут проверять вводимые в форму данные. В случае обнаружения ошибок, форма покажет пользователю сообщение для конкретной ошибки. К примеру, в случае, если пользователь не нажал “Согласиться с правилами” или ввел неверный пароль в графе “Подтвердите пароль” или некорректный email адрес / номер телефона и т.д.

Правила валидации

Для управления правилами валидации атрибутов сущности, откройте диаграмму классов (пункт меню “Модель”) и выберите класс нужной сущности. В правой панели в Настройках сущности (доступна только в режиме редактирования) откройте раздел “Валидация”. Здесь вы можете создать для сущности новые правила валидации атрибутов, удалять или изменять существующие.

Настройка валидации на экранной форме

Компонент “Статус валидации” привязывается к форме и сущности через параметры и имеет 3 возможных состояния:

- Валидация прошла успешно
- Валидация в процессе
- Валидация прошла с ошибками (отображается список ошибок)

Чтобы посмотреть валидацию сущности в действии, нужно в Компонентах добавить на страницу и настроить компонент “Статус валидации”, проще всего это можно сделать, нажав на кнопку “сгенерировать UI” на странице “Модель” в “Настройках сущности”. Это действие генерирует компонент формы со встроенным компонентом “Статус валидации”, привязанным к сущности.

Форма валидна

Client

Город	▼	firstName Сеня	Фамилия Петров
-------	---	-------------------	-------------------

✓ Ошибок не найдено

Отмена Сохранить

Форма валидна

Форма невалидна

Client

Город



firstName

Фамилия



Ошибка

Укажите имя

Отмена

Сохранить

Форма невалидна

Пример правил валидации данных

- Обязательное поле (not null, not empty)
- Минимальная длина строки
- Максимальная длина строки
- Длина строки (должна быть 3)
- Является ли введенная строка корректным email адресом?
- Минимальное значение числа (например, возраст от 15 лет)
- Максимальное значение числа (сумма до 100.000.000)

Экранные формы (Компоненты)

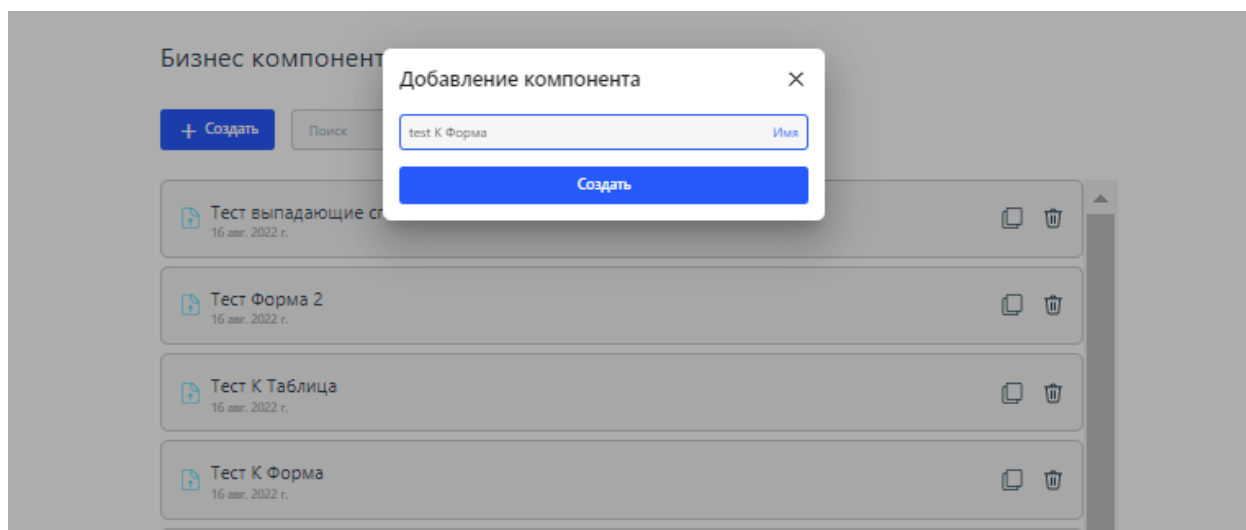
Следующим важным модулем платформы является дизайнер пользовательских интерфейсов. Гибкий и мощный дизайнер и богатая палитра готовых виджетов, позволяют легко создать экранную форму. Дизайнер поддерживает следующие возможности:

1. Готовый набор визуальных компонент из коробки с возможностью добавлять собственные компоненты
2. Удобная визуальная привязка компонент к данным
3. Настройка входных параметров форм и расчетных параметров, включая вызов кастомных методов и SQL запросов.
4. Использование JavaScript в настройках формы для обеспечения её гибкости
5. Переиспользование (наследование) ранее разработанных форм
6. Вызов разработанных форм в других приложениях через iframe

Создание и редактирование бизнес-компонента

На вкладке «**Бизнес-компоненты**» для создания нового компонента необходимо:

1. На главном экране с перечнем бизнес-компонентов нажать на кнопку Создать
2. В открывшемся окне «**Добавление компонента**» заполнить поле «**Имя**» нужным названием
3. Нажать на кнопку Создать
4. Для отмены всех действий нажимаем на кнопку X



Создание

5. Для перехода к режиму редактирования бизнес-компонента, его нужно выбрать из перечня существующих.
6. Чтобы быстро найти нужный бизнес-компонент можно воспользоваться полем Поиск.

7. Для создания копии ранее созданного бизнес-компонента необходимо нажать на иконку Копирование
8. Для удаления ранее созданного бизнес-компонента необходимо нажать на иконку Корзина (См. рисунок ниже).

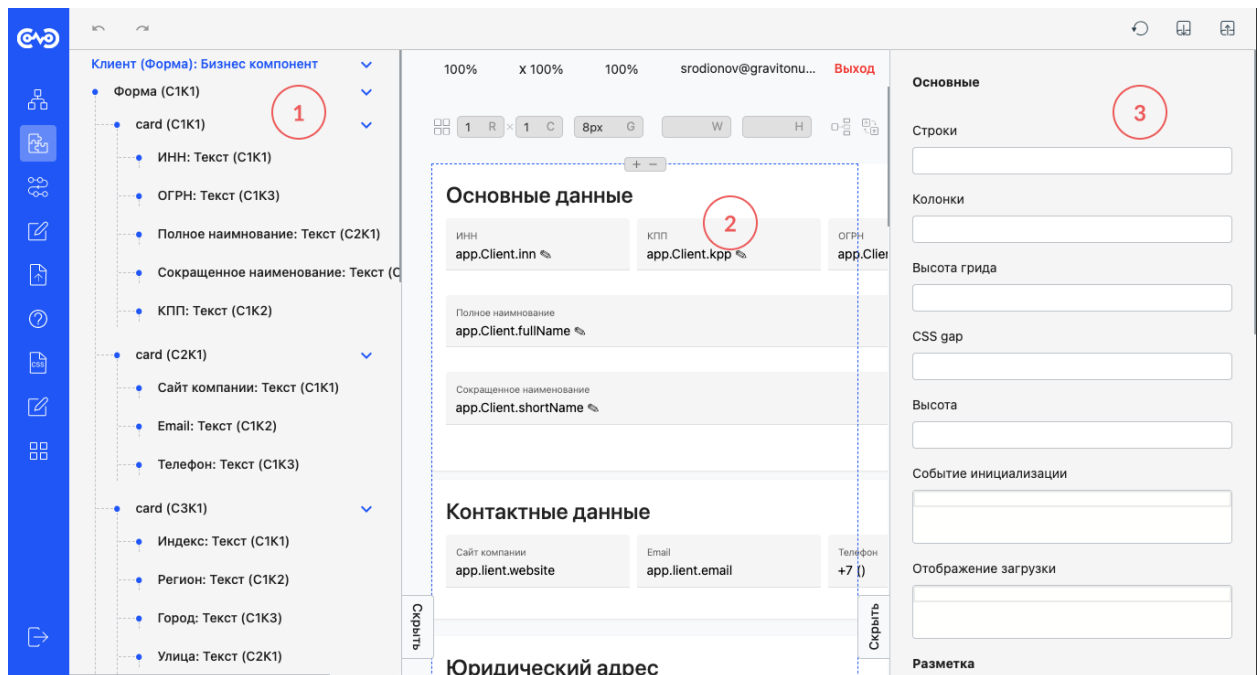
Бизнес компоненты



Создание

Работа с дизайнером интерфейсов

Главный экран дизайнера интерфейсов выглядит как показано на рис ниже



общие положения

На главном экране дизайнера интерфейсов выделены 3 главных блока:

Навигация (Блок 1)

В данном блоке отображается информация о том, какие виджеты находятся на форме. Информация обновляется сразу после добавления очередного компонента на форму. Так же

через нее можно выбрать тот или иной компонент, который необходим для изменения его параметров.

Канвас (Блок 2)

В данном блоке происходит непосредственно проектирование внешнего вида формы

Сетка расположения виджетов

Если вы планируете, что ваша форма будет подтягивать данные при открытии и сохранять при нажатии кнопки сохранить, то нужно задать сетку размером 1 x 1, и первым элементом указать виджет “Форма”.

Сетка для расположения виджетов может быть задана на следующих элементах:

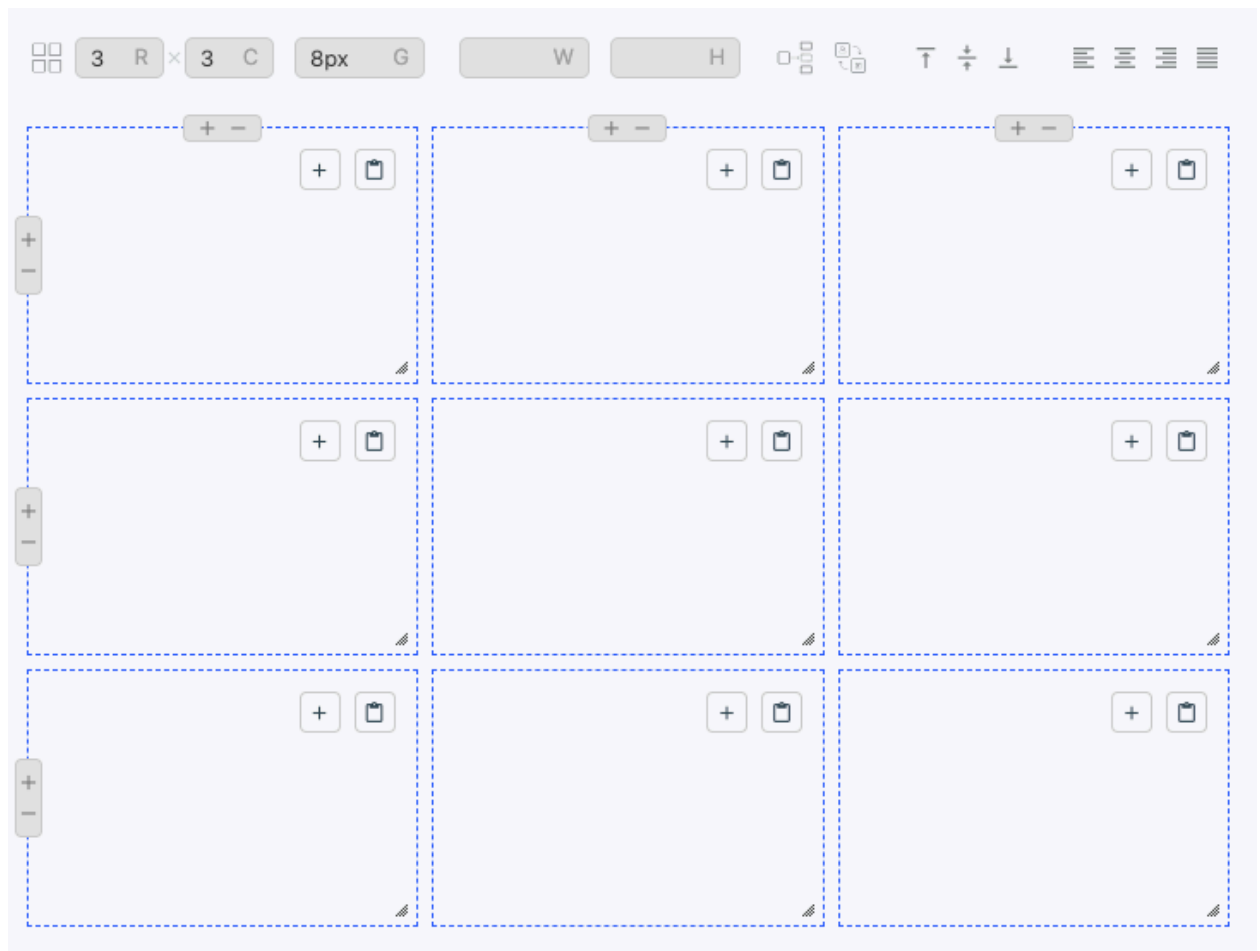
- Бизнес-компонент
- Форма
- Карточка
- Контейнер

Для того, чтобы задать сетку, необходимо указать количество строк и столбцов в блоке



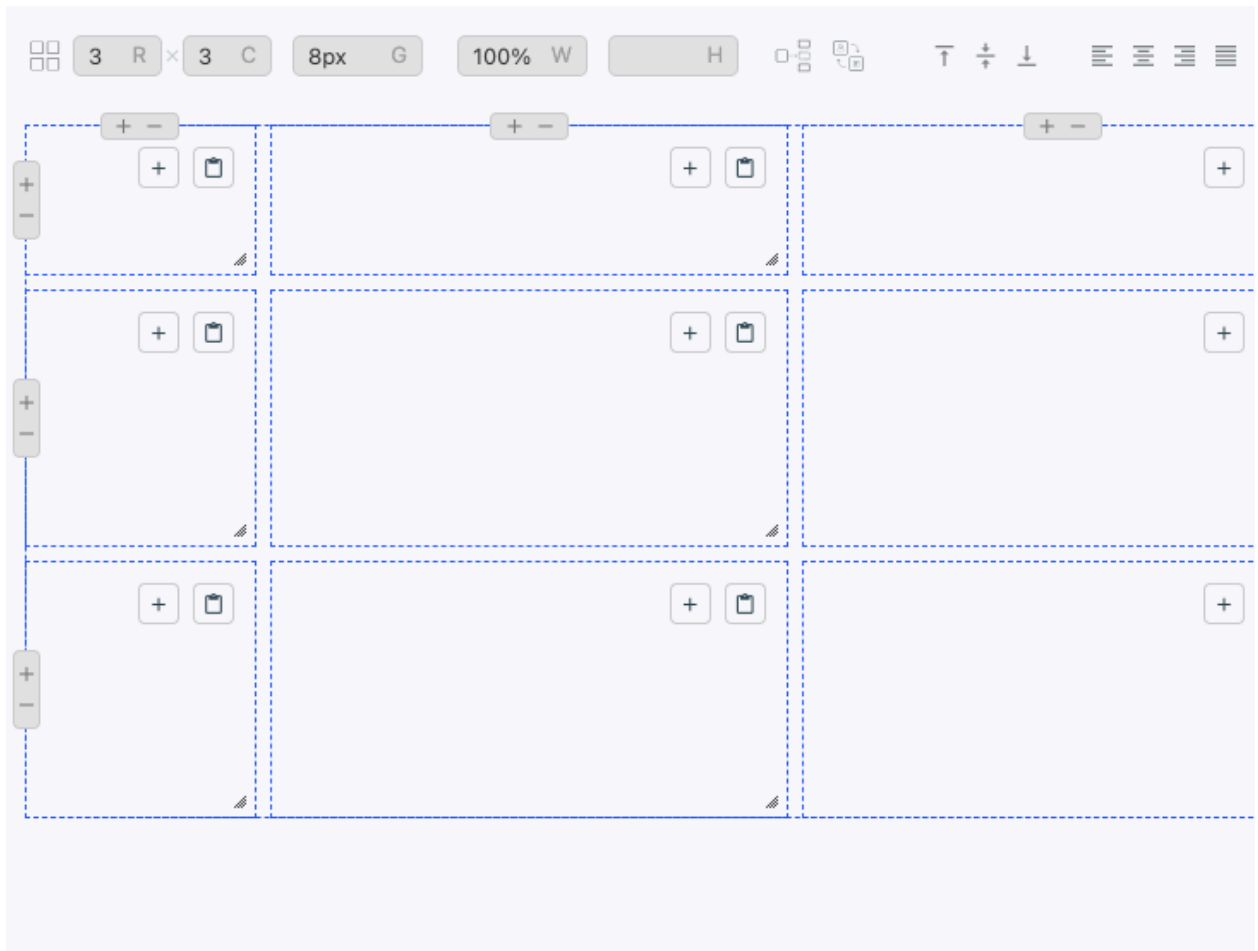
Сетка

Например, если вы зададите размер 3 x 3, то сетка будет выглядеть так



Если вы хотите поменять ширину или высоту столбцов, встаньте в меню слева на компонент с сеткой (например Бизнес-компонент), после того как на канвасе отобразится сетка, подведите курсор мыши к синим линиям столбцов или строк и после появления значка стрелки, нажмите правую кнопку мыши и тяните в соответствующую сторону для изменения размера.

Результат выглядит например так



Добавление компонент на сетку

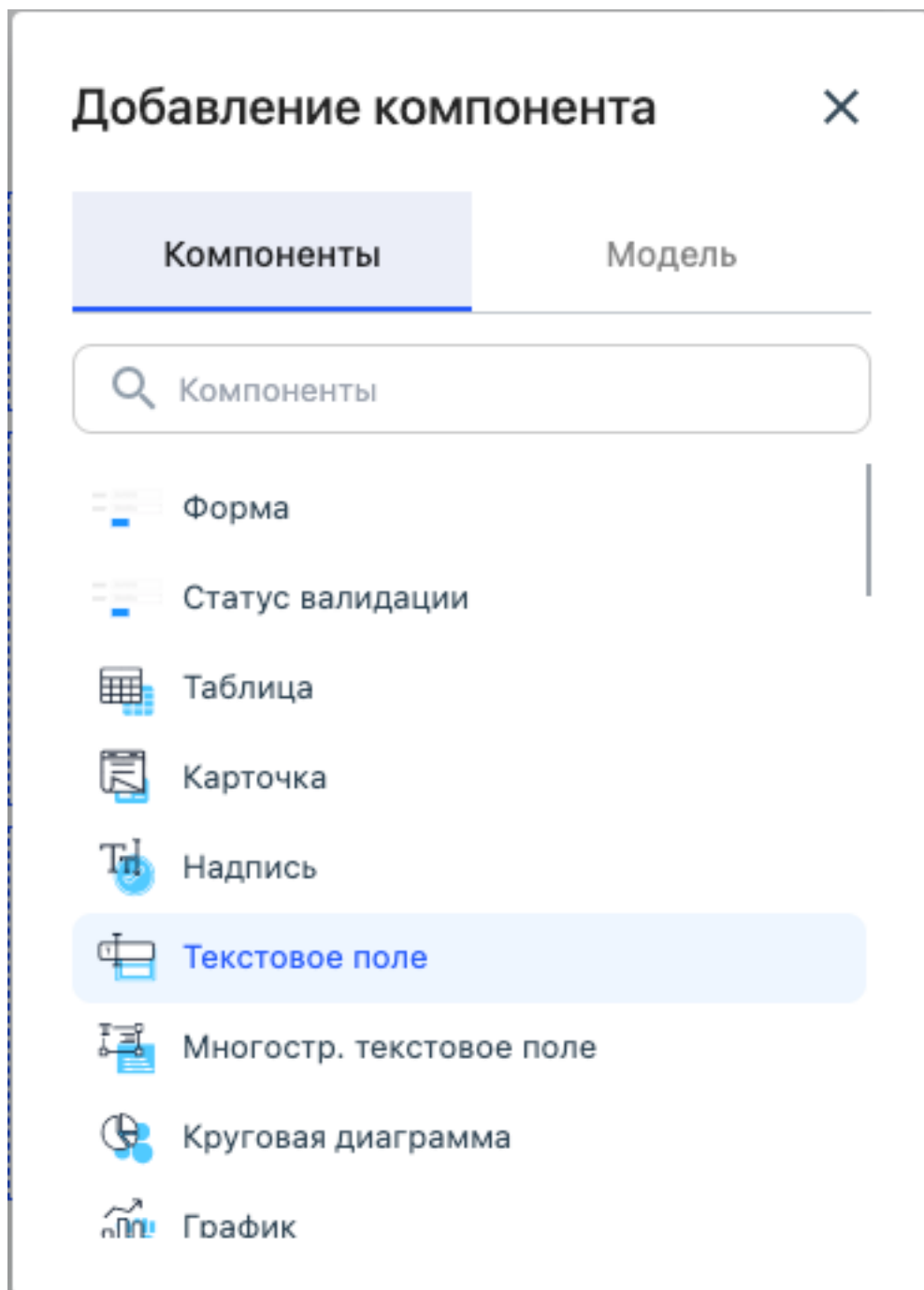
Для того, чтобы добавить компонент на сетку, встаньте в меню слева на компонент с сеткой (Бизнес-компонент\Форма\Карточка\Контейнер) и на выбранной ячейке нажмите значок



Выбор виджета из палитры компонент


Платформа обладает широким набором виджетов для различных целей. Наиболее используемые компоненты это Текстовое поле, Дата, Выпадающий список, Чекбокс, Радиокнопки и кнопки формы.

Выберите необходимый виджет из списка

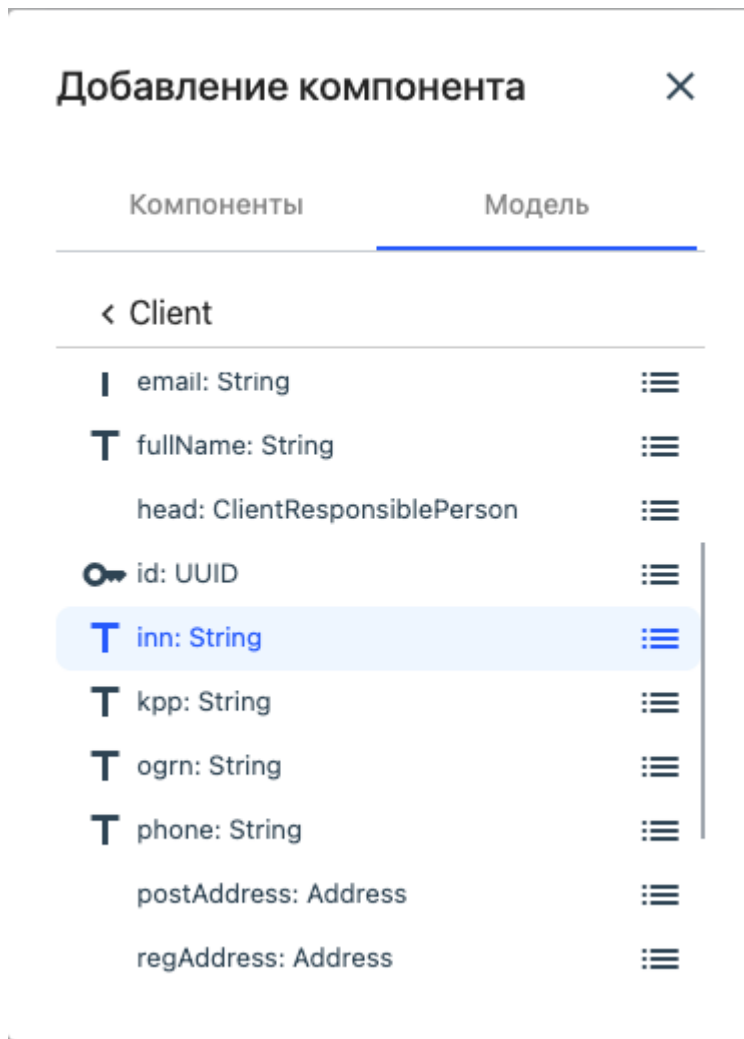


Подбор виджета на основе атрибута сущности

Если у вас уже есть виджет «Форма», то вам доступен подбор виджета на основе атрибута

сущности. На выбранной ячейке нажмите значок  и переключите закладку на “Модель”.

Выберите необходимую сущность и необходимый атрибут, после чего будет добавлен компонент на основании типа атрибута, а также произойдет привязка компонента к модели данных.

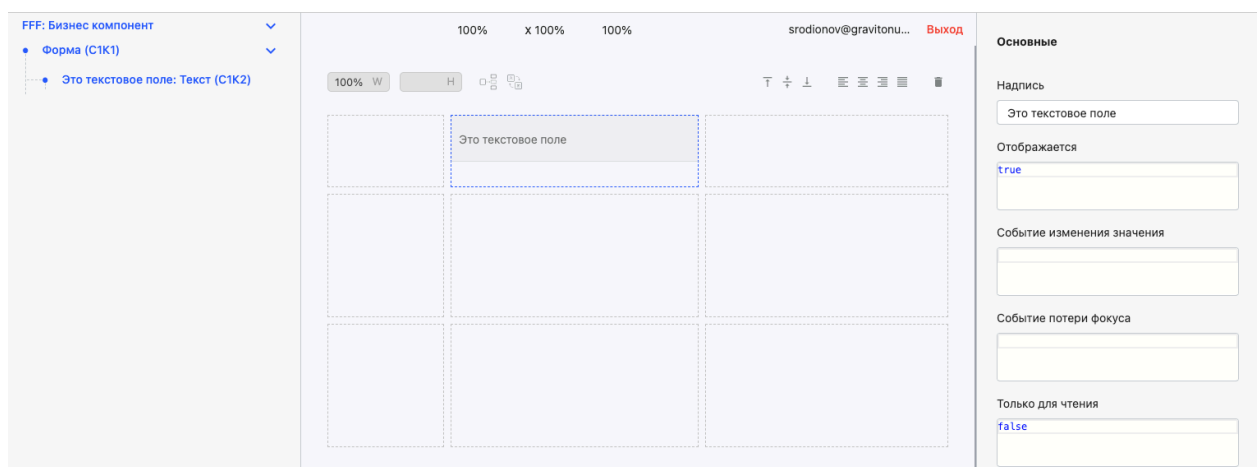


Параметры виджета (Блок 3)

Этот блок предназначен для привязки виджета к модели данных и настройки свойств виджета.

Выберите слева в Навигации виджет для настройке или наведите на него курсор мыши и нажмите левую кнопку.

На правой панели свойств отобразятся данные именно для данного виджета



У каждого виджета свой набор свойств, при этом часть свойств является общей для большинства компонент.

- **Надпись** - это тест, который появляется на компоненте
- **Отображается** - это свойство, которое показывает будет ли показываться виджет на форме, принимает значения true и false
- **Только для чтения** - это свойство, которое показывает будет ли доступен виджет для редактирования, принимает значения true и false
- **Содержимое подсказки** - это текст, который появляется при наведении курсора мыши на виджет, текст вводится в одинарных кавычках
- **Индекс табуляции** - определяет порядок обхода виджетов на форме используя Tab
- **Имя** - уникальное имя виджета, которое позволяет другим виджетам и скриптам обращаться к нему
- **Прочитать из** - указывается GraphQL метод для начитки данных, например `app.clientFindById.name`
- **Сохранить в** - указывается GraphQL метод для сохранения данных, например `app.clientSave.entity.name`

Описание внешних переменных

Для того, чтобы на уровне бизнес-компонента оперировать данными переданными из других форм, существует механизм описания внешних переменных.

Для того, чтобы задать внешнюю переменную, необходимо встать в Навигаторе слева на бизнес-компонент и на панели Свойства справа, прокрутить в самый низ.

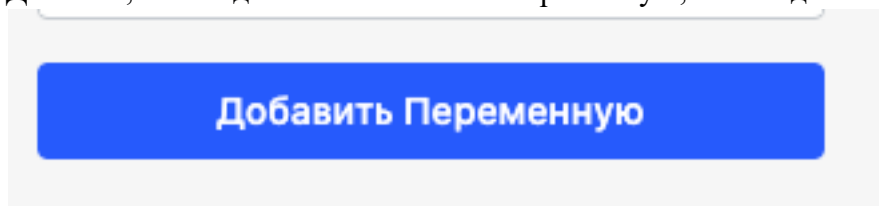
По умолчанию есть две внешних переменных:

1. **roles** типа **Объект**, содержит все роли пользователя
2. **userName** типа **Строка**, содержит логин пользователя

Для того, чтобы открыть в форме значения какой-то конкретной записи сущности, например, Клиента, необходимо передать значение id клиента через внешнюю переменную.

Добавление внешней переменной

Для того, чтобы добавить внешнюю переменную, необходимо нажать кнопку



В открывшемся окне, навести курсор мыши на серую надпись “Название” и кликнув один раз правой кнопкой мыши ввести название переменной.

Если тип переменной не **Строка**, нужно кликнув на **Строка** выбрать другой тип из списка.

Устанавливаем чек бокс “Внешняя переменная”

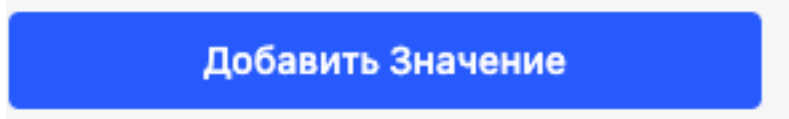
Можно добавить вычисляемую переменную, не указывая, что это внешняя переменная, указав запрос в белом поле под названием переменной.

Создание значений по умолчанию

Для работы формы, генерируются запросы вида `app.clientFindById` с переданным ему внешним параметром `id`, в список полей которого войдут:

- все поля, перечисленные в виджетах в свойстве “Прочитать из”, например `app.clientFindById.inn` добавит в список полей запроса `inn`
- все значения по умолчанию с признаком “Добавить поле в запрос”.

Добавим переменную для генерации запроса нажав кнопку



В открывшемся окне, навести курсор мыши на серую надпись “Название” и кликнув один раз правой кнопкой мыши ввести название переменной. Если тип значения не Строка, нужно кликнуть на Строка выбрать другой тип из списка.

- Установка чек бокса “Добавить поле в запрос” добавляет в список возвращаемых полей данное поле
- Установка чек бокса “Добавить поле в параметры” добавляет поле в параметры запроса.

Работа с формулами в свойствах виджетов

Ниже представлены некоторые формулы, которые можно использовать в свойствах виджетов:

Объединение нескольких полей в одно

Объединение полей осуществляется конкатенацией “+” нескольких полей.

В качестве переменных могут быть как имена других виджетов,

```
(fam || "") + " " + (imya || "") + " " + (otch || "")
```

так и данные запросов

```
(app.employeeFindById.lastName || "") + " " + (app.employeeFindById.firstName || "") + " " + (app.employeeFindById.middleName || "")
```

Изменение регистра

Функции `toLowerCase()` / `toUpperCase()` позволяют изменить регистр значения, например `app.employeeFindById.lastName.toUpperCase() \\ ИВАНОВ`

Перевод на другую страницу

Данная функция используется как постобработчик для кнопок формы, например, после нажатия кнопок Сохранить или Отмены нужно вернуться на общий список клиентов `routerLink('/client/all')`, где `'/client/all'` это URL формы в бизнес приложении.

Интервал

В виджетах типа Дата или Дата и время, часто бывает необходимо вычислить интервал от текущей даты или от даты рождения, для задания ограничений

```
app.CV.candidate.birthDate ?  
DateTime.fromISO(app.CV.candidate.birthDate).diffNow(['years', 'month']).negate().years : '-'  
,
```

Бизнес-процессы

В состав платформы входит редактор бизнес-процессов, поддерживающий нотацию BPMN 2.0 ([wiki](#))

Нотация BPMN состоит из условных обозначений для отображения бизнес-процессов в виде диаграмм. Нотация ориентирована как на технических специалистов (разработчиков, ответственных за реализацию процессов), так и на бизнес-пользователей (бизнес-аналитиков, создающих и улучшающих процессы) и менеджеров, следящих за процессами и управляющих ими. Неоспоримым плюсом этой нотации является ее простота и гибкость. Простота достигается за счет наглядности — условные обозначения, используемые для отображения бизнес-процессов, описаны в виде диаграмм и блок-схем. Гибкой ее делает набор элементов и правил, с помощью которых построить бизнес-процесс даже рядовой пользователь

Дизайнер бизнес-процессов платформы обладает следующими возможностями: 1. поддержка русского и английского языка интерфейса 1. поддержка нотации BPMN 2.0 и DMN 1.3 1. импорт/экспорт схем бизнес-процессов и DMN схем 1. проверка бизнес-процессов на соответствие правилам (правила задаются пользователем платформы) 1. деплой бизнес-процесса или схемы в движок бизнес-процессов 1. наличие готовых Java делегатов, для типовых операций, таких как отправка email и sms, загрузка данных с ftp, вызов методов системы Контур-Фокус, и другие.

Смоделированные бизнес-процессы будут запускаться во встроенном движке бизнес-процессов BPMN Engine (Camunda).

Программные продукты класса BPM engine – нацелены на реализацию только управления процессами. Могут использоваться как в качестве классического центрального компонента управления всеми бизнес-процессами, так и встраиваться туда, где это нужно.

Платформа использует Camunda BPM в режиме embedded, что позволяет использовать широкий перечень методов API для работы с движком из бизнес-приложения.

Элементы BPMN нотации

BPMN-процесс – это любой бизнес-процесс, отражённый с помощью нотации.

Процессы состоят из элементов, каждый из которых обозначается на схеме специальным значком.

Элементы нотации BPMN – это элементы графической схемы, но также и элементы самого бизнес-процесса.

Нотация опирается на следующие базовые графические элементы:

1. Объекты потока управления: события, действия и логические операторы (развилки)
2. Соединяющие объекты: поток управления, поток сообщений и ассоциации
3. Роли: пулы и дорожки
4. Артефакты: данные, группы и текстовые аннотации.

Элементы этих четырёх категорий позволяют строить простейшие диаграммы бизнес-процессов. Для повышения выразительности модели спецификация разрешает создавать новые типы объектов потока управления и артефактов.

События

В событиях можно выделить следующие типы: 1. начальное, 1. промежуточные, 1. конечное.

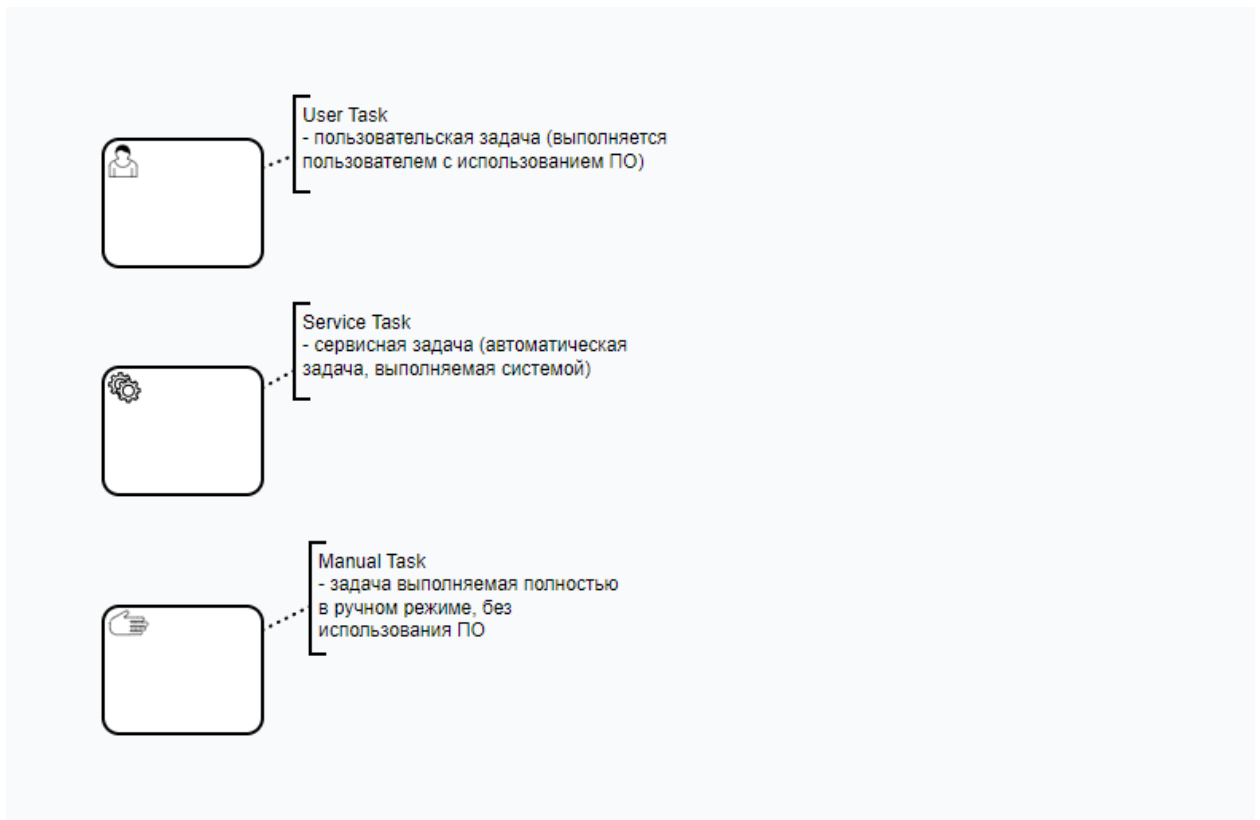
Ниже на рисунке приведены несколько вариантов этих событий.



Задачи

Основные типы элемента **Задача**:

- User Task - пользовательская задача (выполняемая пользователем с использованием ПО),
- Service Task - сервисная задача (автоматическая задача, выполняемая системой),
- Manual Task - задачи, выполняемые пользователем вручную.



Шлюзы

Основные типы элемента Шлюз:

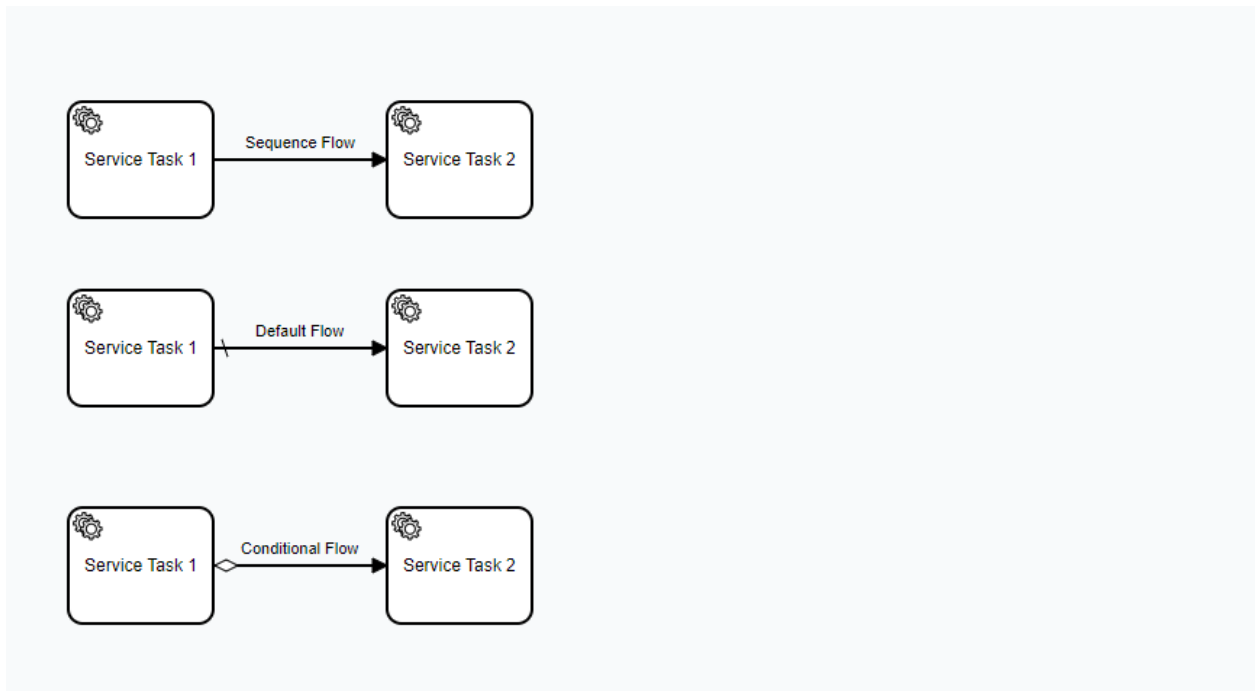
- Parallel Gateway - параллельный шлюз, используется для обозначения слияния/ветвления потоков управления в рамках процесса,
- Exclusive Gateway - эксклюзивный (исключающий) шлюз, используется для ветвления потока управления на несколько альтернативных потоков, когда выполнение процесса зависит от выполнения некоторого исключаяющего условия,
- Inclusive Gateway - неэксклюзивный шлюз, используется для ветвления потока управления на несколько потоков, когда выполнение процесса зависит от выполнения условий,
- Event based Gateway - эксклюзивный шлюз по событиям, используется для ветвления потока на несколько альтернативных потоков, когда дальнейшее выполнение процесса зависит от возникновения некоторого события-обработчика, следующего после шлюза. Событие, идущее после шлюза и возникшее первым, определяет дальнейший ход выполнения процесса.



Поток управления

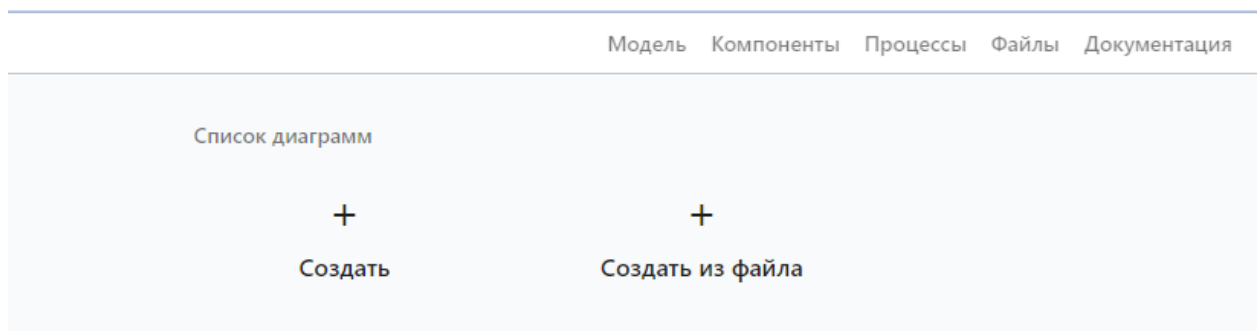
Основные типы элемента Поток управления

- Sequence Flow - обычный поток управления,
- Default Flow - поток управления по умолчанию, используется, когда необходимо показать, что выполнение процесса будет происходить по этому потоку только если не выполняется ни одно из заданных условий,
- Conditional Flow - условный поток управления, используется чтобы показать, что выполнение процесса будет происходить по этому потоку только в том случае, когда выполняются заданные условия. Такой тип элемента выбирается, если условный поток управления является исходящим от процесса.



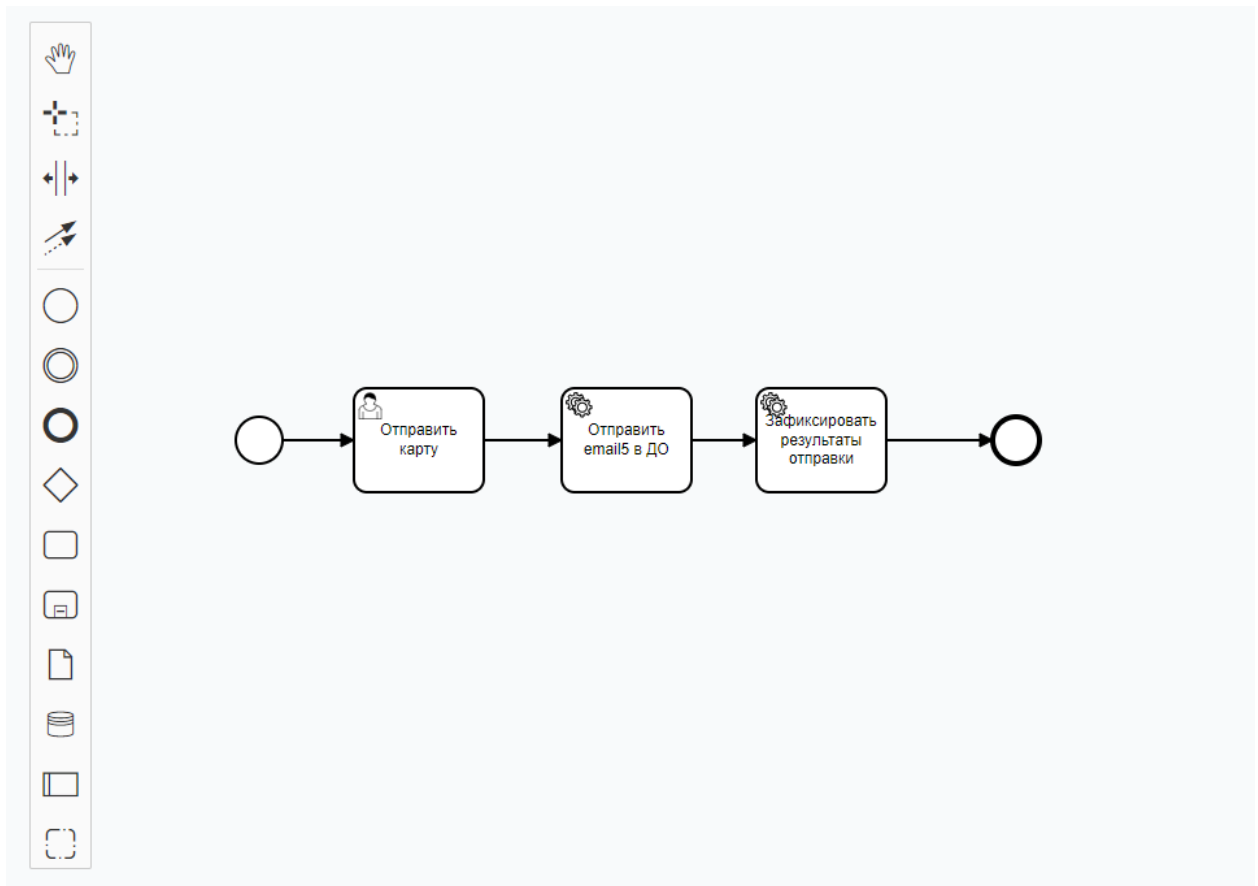
Создание бизнес-процесса

Новый бизнес-процесс создается по нажатию Создать на странице “Процессы”

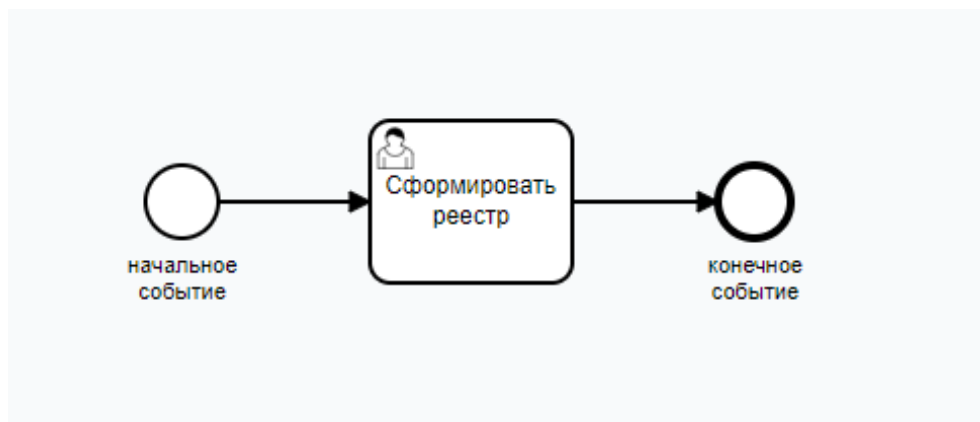


По нажатию на Создать из файла можно загрузить созданный ранее бизнес-процесс, доступный в виде файла .brmn

При создании бизнес-процесса необходимые элементы выбираются на Панели элементов, расположенной в левой части, и соединяются потоками управления.



Любой бизнес-процесс начинается с начального события и заканчивается конечным событием.



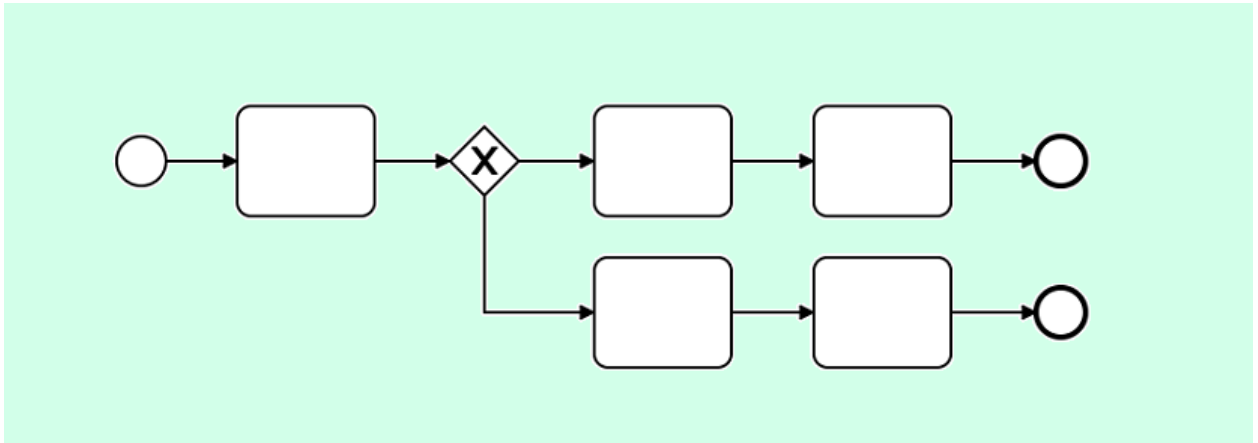
Стартовое событие указывает на то, в какой точке берет начало тот или иной процесс. Оно является начальной точкой в процессе, никакой входящий поток не может быть соединен со стартовым событием.

Стартовое событие в нотации BPMN изображается в виде круга со свободным центром.

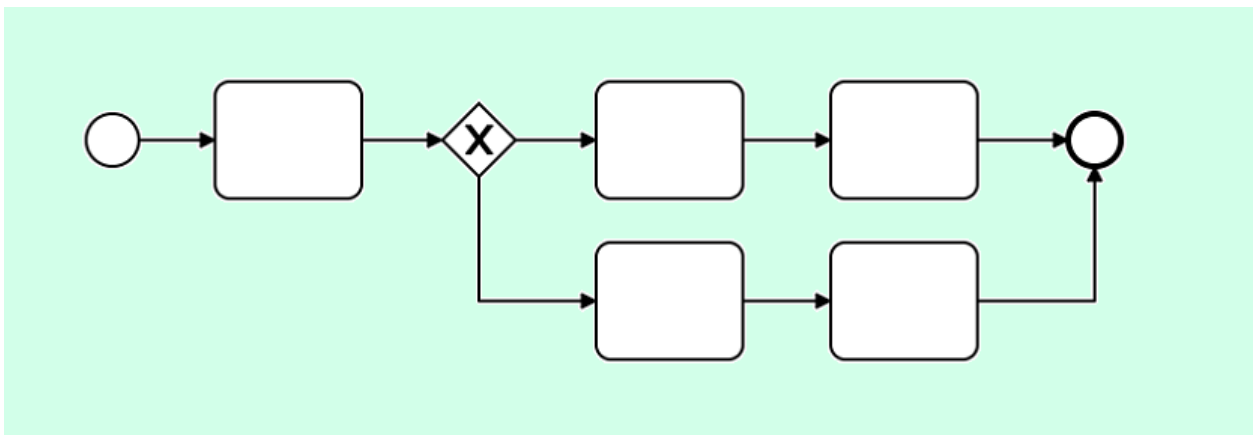
Конечное событие указывает на то, в какой точке завершается тот или иной процесс. Оно завершает ход процесса, никакой исходящий поток не может быть соединен с конечным событием.

Конечное событие представляет собой круг, выполненный одиночной, жирной линией.

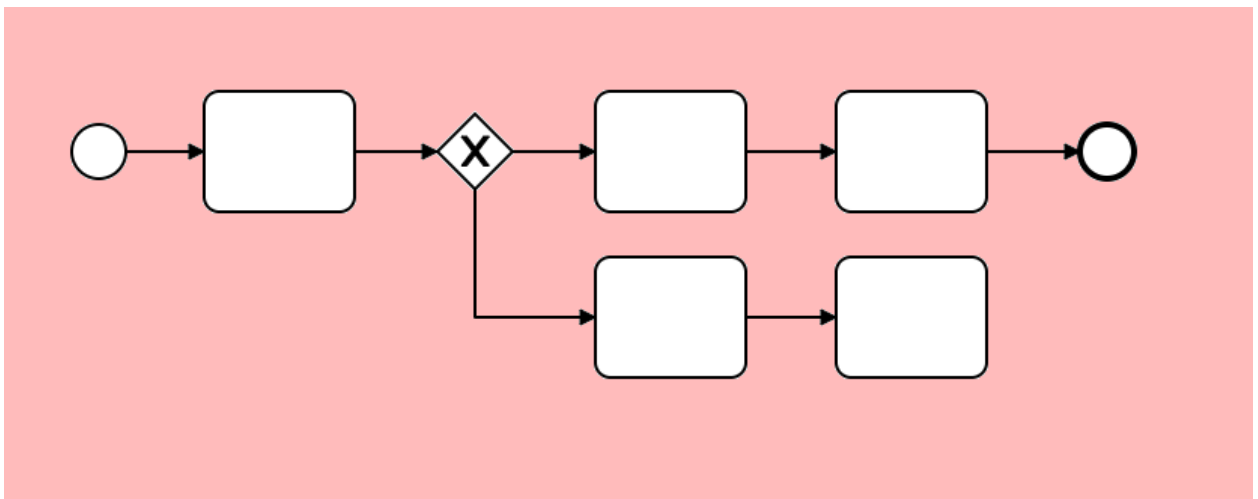
Процессы могут содержать несколько конечных событий.



Конечное событие может быть соединено с несколькими входящими потоками.




Не должно быть задач “повисших в воздухе”. Если процесс заканчивается, ставим конечное событие, как на рисунках выше.





Выделив любой элемент процесса, справа от него появляется панель кнопок.




На панели расположены следующие кнопки:

 создать следующий элемент процесса, связанный с выделенным потоком управления

 добавить текст аннотации к элементу

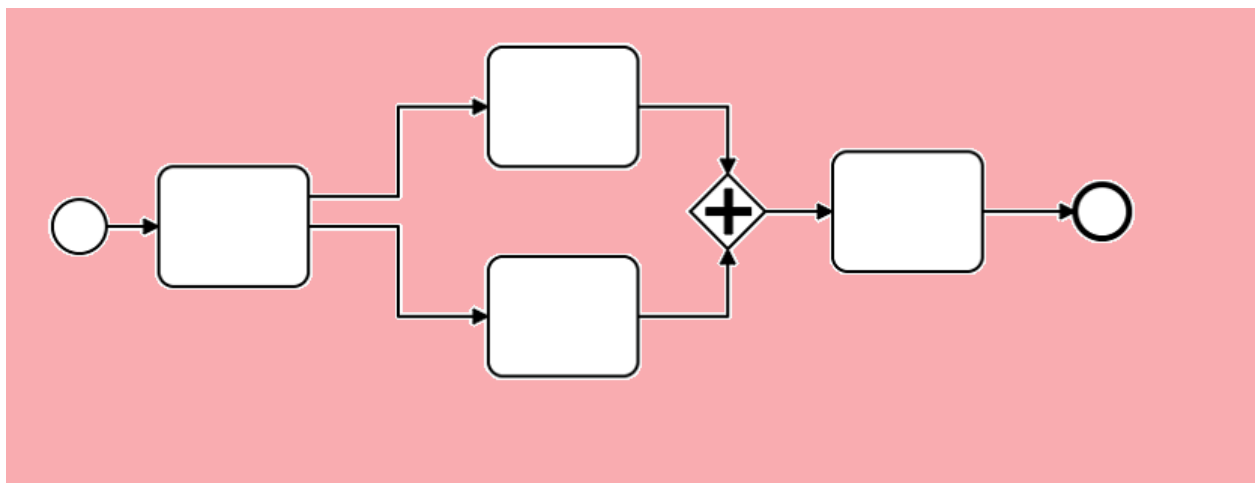
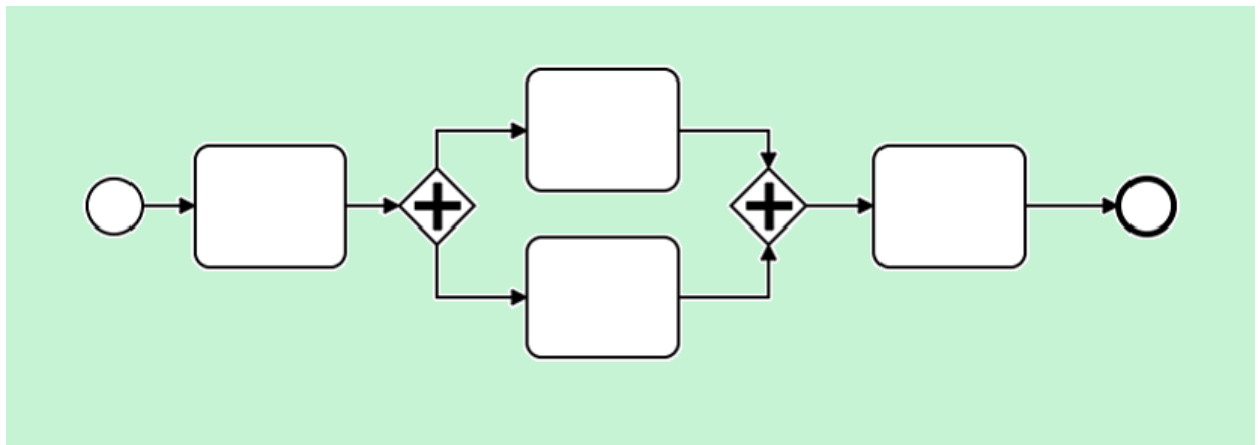
 изменить тип элемента

 удалить элемент

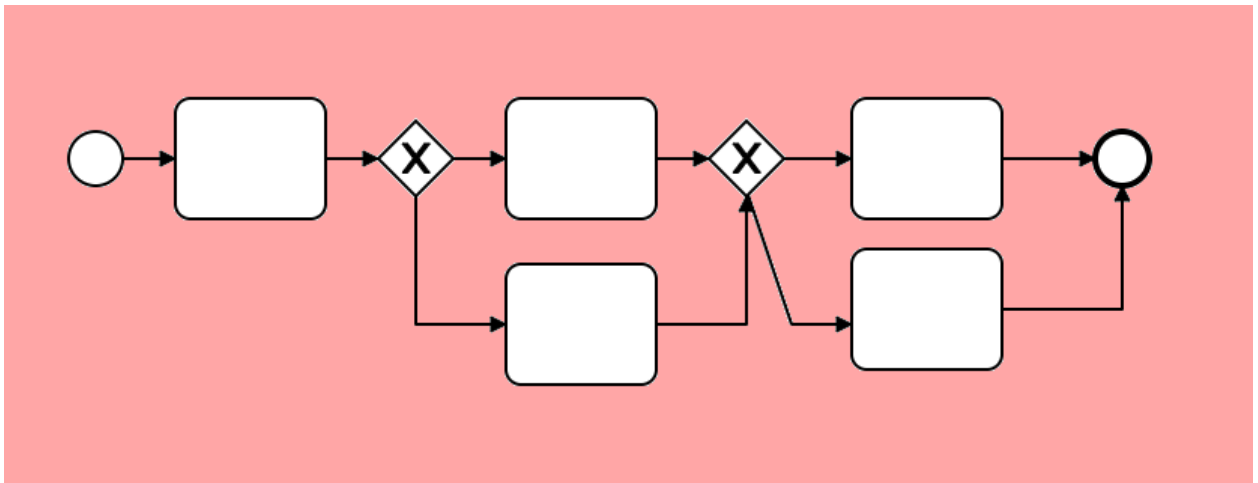
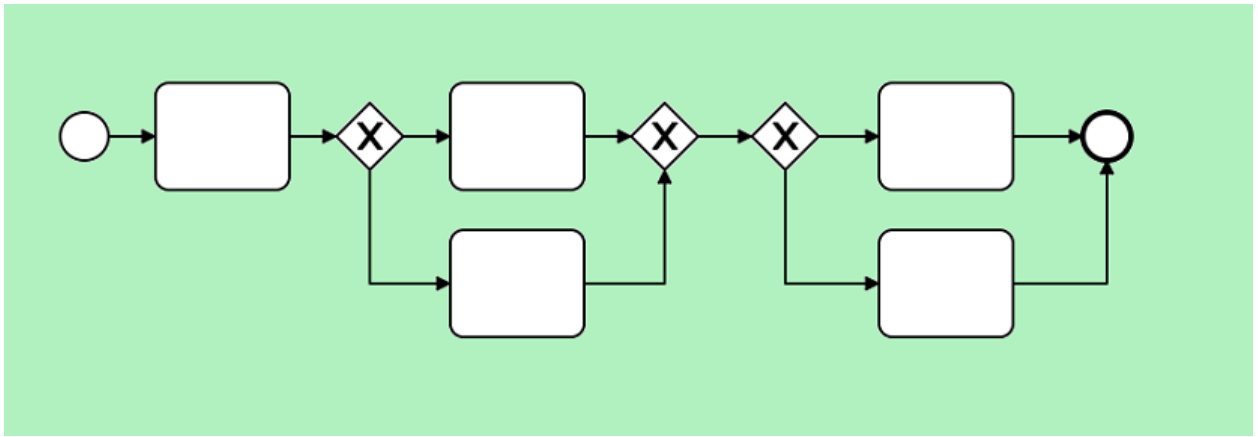
 связать элемент с другим элементом

Поток может идти как по альтернативным, так и по параллельным маршрутам.

Для соединения и разъединения потока используем шлюзы



Один шлюз не может соединять и разъединять потоки.



Создание DMN моделей

Модель принятия решений и нотация (DMN) является отраслевым стандартом для моделирования и принятия решений, которые определяются бизнес правилами.

Диаграммы DMN хорошо дополняют модели бизнес-процессов, созданные в нотации BPMN и могут быть вызваны непосредственно из процесса с помощью компонента Business Rule.

DMN диаграммы (Decision Requirements Graph - DRG) состоят из последовательности шагов, каждый из которых представляет собой один из компонент:

- Input Data - входящие данные
- Decision, элемент может быть представлен или Decision Table или Literal Expression
- Knowledge Model (не автоматизируется)
- Knowledge Source (не автоматизируется)

Decision Table

Decision Table (таблица решений) — техника, помогающая наглядно изобразить комбинаторику условий.

Таблица решений состоит из:

1. Название
2. Метод срабатывания (hit policy). Если там стоит U, то значит у нас политика срабатывания — unique. Т.е. решение должно всегда отдавать одно, уникальное решение (пересечение не допускается). Есть несколько вариантов.

3. Входные переменные.
4. Выходные данные — для каждой возможной входящей записи мы определяем выходную переменную.
5. Правило — это набор входных и выходных данных, строка таблицы. У каждого правила есть номер, он указан в таблице слева.
6. Аннотация — это текст справа, он используется для объяснения правила и не автоматизируется.

Literal Expression

FEEL

Детальная информация по FEEL представлена [тут](#)

Выражения (Expressions)

Boolean expressions

- = - равно
- != - не равно
- < - меньше
- <= - меньше или равно
- > - больше
- >= - больше или равно
- between [x] and [y] - то же самое что (`_ >= x and _ <= y`)

Примеры:

```
5 = 5 // true
```

```
5 != 5 // false
```

```
date("2020-04-05") < date("2020-04-06") // true
```

```
time("08:00:00") <= time("08:00:00") // true
```

```
duration("P1D") > duration("P5D") // false
```

```
duration("P1Y") >= duration("P6M") // true
```

```
5 between 3 and 7 // true
```

```
date("2020-04-06") between date("2020-04-05") and date("2020-04-09") // true
```

String expressions

- + - конкатенация
`"foo" + "bar"; // "foobar"`

Numeric expressions

- + - сложение
- - - вычитание
- / - деление
- * - умножение
- ** - возведение в степень

List expressions

- [1, 2, 3, 4]; create new list
- a[i]; получить элемент списка

Temporal expressions

- `date("2020-04-06")` - задать дату
- `time("08:00:00")` - задать время
- `time("08:00:00@Europe/Berlin")` - задать время и часовой пояс
- `date and time("2020-04-06T08:00:00")` - задать дату и время
- `duration("P1Y6M")` - задать длительность

Примеры:

```
date("2020-04-06") + duration("P1D") > // date("2020-04-07")
```

```
date("2020-04-06") - date("2020-04-01"); // duration("P5D")
```


```
duration("P1D") * 5; // duration("P5D")
```

```
date("2020-04-06").year // 2020
```

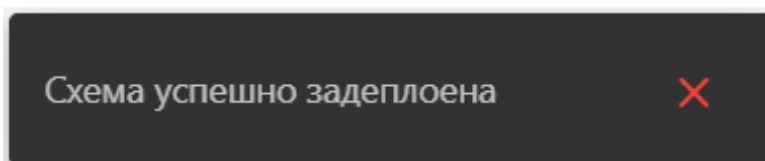
Деплой диаграммы на сервер

Для того, чтобы задеплоить созданную DMN схему, необходимо сохранить диаграмму,

нажав кнопку Save  вверху справа.

После этого нажать кнопку Deploy 

В случае успешного выполнения операции, появится сообщение



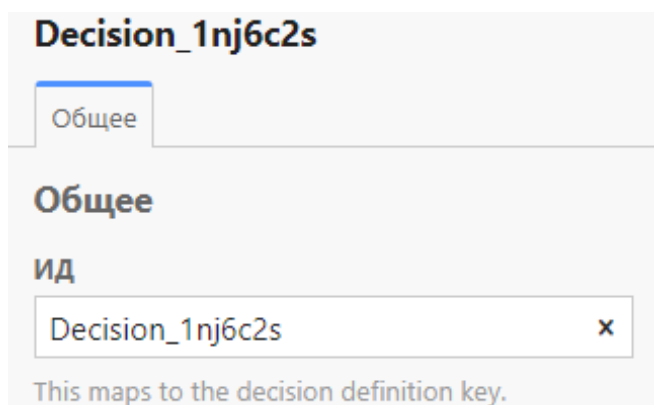
либо сообщение об ошибке.

Как вызвать DMN через REST

Для того, чтобы вызвать DMN правило через REST, необходимо вызвать метод

POST `[host]/camunda-api/decision-definition/key/[definition key]/evaluate`

где `definition key` - id вызываемого бизнес-правила (! если у вас используется граф, то нужен id самого последнего блока)



Настройка ServiceTask

Активность с типом ServiceTask - это автоматическая задача выполняемая системой.

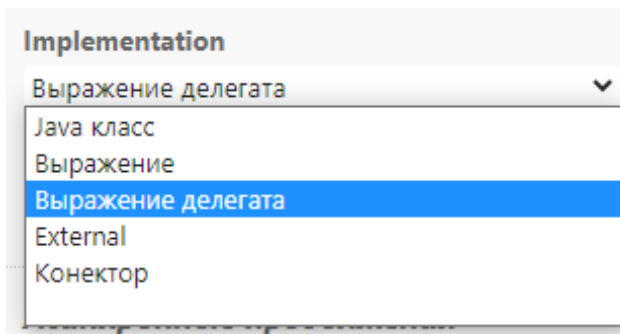
В диаграмме обозначается символом шестеренки



ServiceTask

После добавления ServiceTask необходимо справа на панели свойств заполнить обязательные поля:

- **Имя** - Наименование сервис таска отображаемое на диаграмме
- **Имплементация**



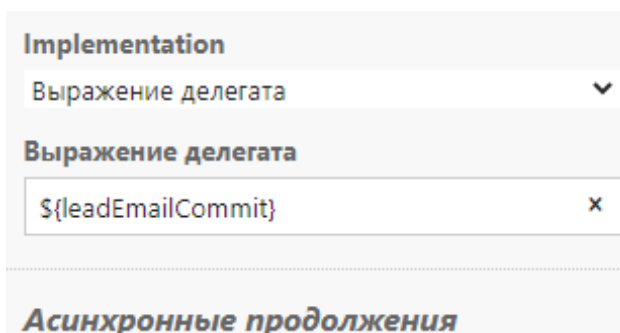
Есть следующие имплементации

- Java класс - Выражение
- Выражение делегата
- External - Коннектор

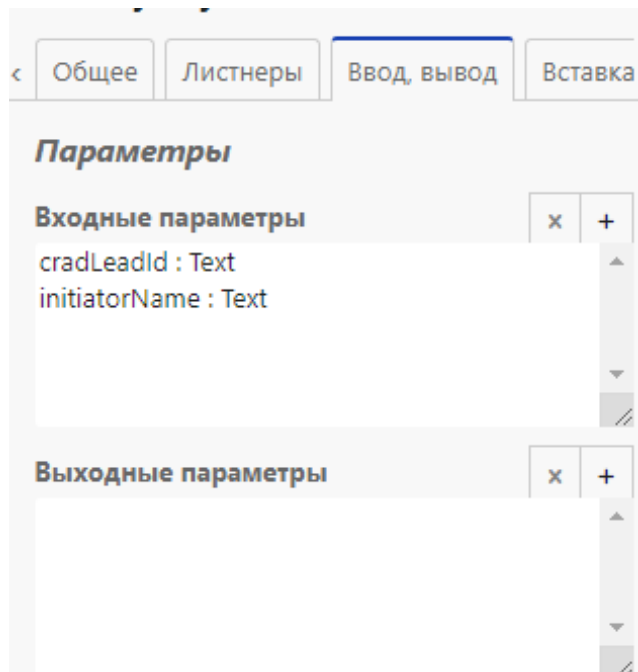
Мы рекомендуем использовать тип “Выражение делегата”

- **Выражение делегата**

Это наименование Java делегата.



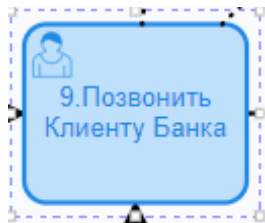
Если необходимо передать в ServiceTask параметры из процесса в качестве входящих переменных или сохранить в процесс итоговые данные, воспользуйтесь закладкой “Ввод, вывод”



Настройка UserTask

Активность с типом UserTask - это пользовательская задача.

В диаграмме обозначается символом человека

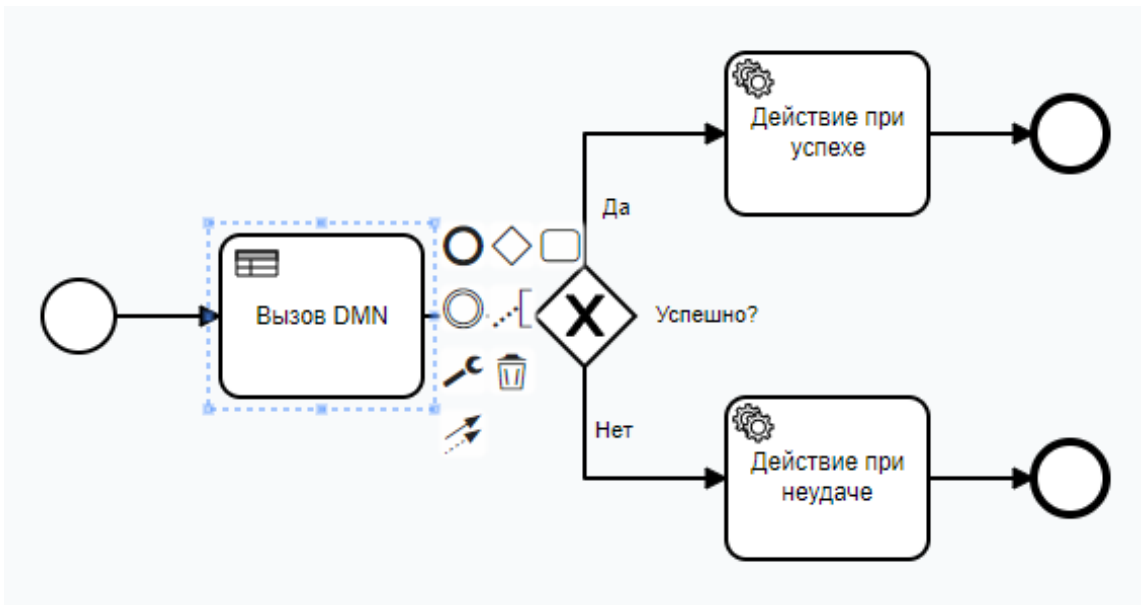


После добавления UserTask необходимо справа на панели свойств заполнить обязательные поля:

Имя - Наименование юзер таска отображаемое на диаграмме

Настройка бизнес-таска

Для того, чтобы вызвать в процессе ранее задеплоенную DMN диаграмму, нужно в процессе добавить активность с типом “Выполнение бизнес-правила”



В настройках активности указать:

- Имплементацию

Implementation
 DMN

- Ссылку на id задеплоенного DMN правила (! если у вас используется граф, то нужен id самого последнего блока)

Ссылка на решение
 scoring

- Наименование результирующей переменной

Результирующая переменная
 Score

- Тип результирующей переменной

Результат карты принятия решения
 singleEntry (TypedValue)

Если необходимо передать в бизнес-правило параметры из процесса в качестве входящих переменных или сохранить в процесс итоговые данные, воспользуйтесь закладкой “Ввод, вывод”

Json трансформация

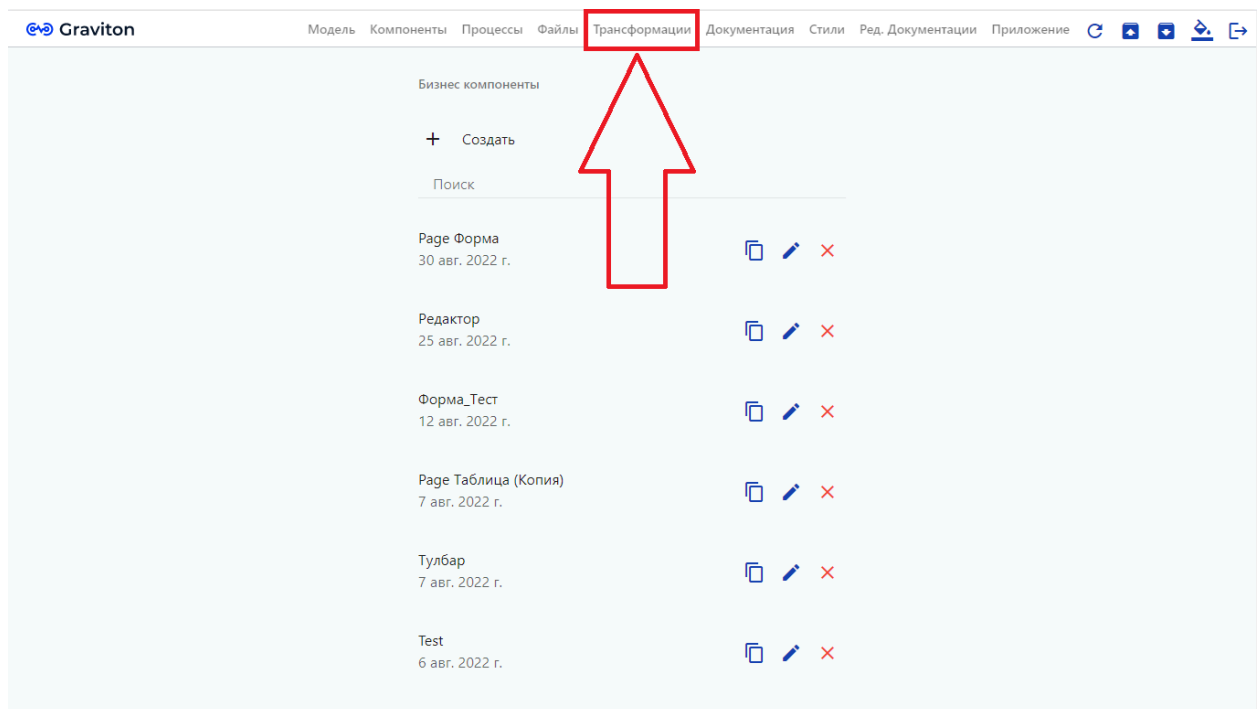
Json трансформация используется для того, чтобы преобразовывать данные, полученные из других сервисов в структуру данных бизнес-приложения.

К примеру, сервис запрашивает данные о количестве, например, роз на складе. Сервис вызывает метод в другой программе/приложении для получения данных. Json трансформация, получив данные, сохраняет полученные данные в места, которые были ей указаны.

Как перейти в трансформации?

В трансформации можно перейти через приложение, если знать url, или через платформу. Рассмотрим переход через платформу приложения.

В платформе приложения необходимо в тул-баре выбрать вкладку “Трансформации”, которая ведет на компонент.



Graviton						
Модель Компоненты Процессы Файлы Трансформации Документация Стили Ред. Документации Приложение						
<input type="text" value="Найти"/>						
НАЗВАНИЕ	КОД	SOURCE TYPE	S	DESTINATION TYPE	КОММЕНТАРИЙ	
No data to display						
total						

Как работать с json трансформацией?

В приложении JsonTransform выглядит как обычная таблица, которая имеет:

1. Название - название json трансформации
2. Код - код трансформации
3. Источник - источник, откуда поступают данные, обычно внешний сервис
4. Версия источника - версия используемого источника
5. Получатель - полное название сущности, в которую будут сохраняться данные после трансформации
6. Описание - описание трансформации

Добавление трансформации

7. Нажать на кнопку **“Добавить”** в шапке таблицы

8. В появившейся строке нажать на “-”, откроется форма для редактирования

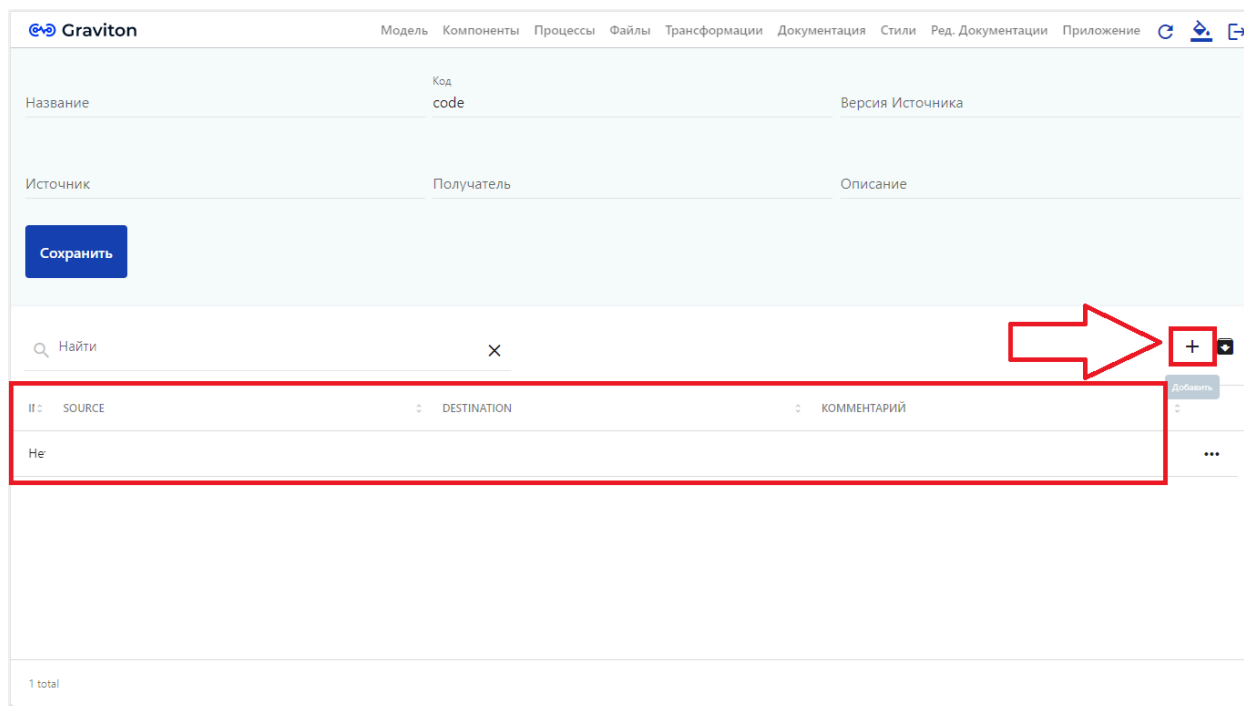
The screenshot shows the Graviton application interface. At the top, there is a navigation bar with the Graviton logo and several menu items: Модель, Компоненты, Процессы, Файлы, Трансформации, Документация, Стили, Ред. Документации, Приложение. Below the navigation bar is a search bar with the text 'Найти' and a close button 'X'. To the right of the search bar is a '+' icon and a 'Добавить' button. Below the search bar is a table with the following columns: НАЗВАНИЕ, КОД, SOURCE TYPE, S, DESTINATION TYPE, КОММЕНТАРИЙ. The first row of the table is highlighted with a red box. The first column contains a hyphen '-', and the second column contains the text 'code'. Below the table is a blue button labeled 'Сохранить'. A red arrow points from the '+' icon to the right, and another red arrow points from the 'code' field to the left. Below the table is another search bar with the text 'Найти' and a close button 'X'. To the right of the search bar is a '+' icon and a 'Добавить' button. Below the search bar is a table with the following columns: ИС, SOURCE, DESTINATION, КОММЕНТАРИЙ. The table is empty, and the text 'No data to display' is shown below it. At the bottom of the page, the text '0 total' is visible.

9. Заполнить поля:

1. Название
2. Код
3. Версия источника
4. Источник
5. Получатель
6. Описание

10. Нажать “Сохранить”, дождаться всплывающего окна о сохранении данных

Добавление параметров трансформации



1. Нажать на кнопку “Добавить” в шапке таблицы, которая идет ниже кнопки “Сохранить” 2. Заполнить Источник (Source), Место сохранения (Destination), Включается ли параметр в трансформацию (Included), Комментарий > Для редактирования Источника, место сохранения и включен ли в запрос этот параметр необходимо двойным кликом активировать поле редактирования в табличке. Другим способом активировать редактирование не получится.

3. Нажать на кнопку “Сохранить”, дождаться всплывающего окна о сохранении данных

Правила Json трансформации

Направлены на работу с одним атрибутом, в декларативном стиле. Создаются на упрощенном jq.

- Доступ к вложенному атрибуту пишется через точку ‘.’. Например:

```
client.firstName, data.result.address.street
```

Массивы

- Для трансформации всех элементов одного массива в другой массив после имени массива используются квадратные скобки, стоящие рядом. Например:

```
data.Employees[].FIO -> department.empployees[].fullName.
```

Иногда на вход подается массив объектов, в этом случае также используем квадратные скобки, только без имени массива впереди.

```
[],EMPLOEES.FIO -> [].employees.fullName
```

- Если из исходного массива надо выбрать 1 элемент по порядковому номеру, то используем квадратные скобки с индексом элемента внутри: [0], [1], [2] и т.д. Нумерация элементов массива начинается с нуля 0. Например:

```
[0].EMPLOEES.FIO -> manager.fullName
```

- И наоборот, можно взять атрибут и положить его в элемент массива.

Data.Manager.FIO -> employees[0].fullName

Data.Manager.BIRTH_DATE -> employees[0].birthDate

- Для поиска в массиве какого-то определенного элемента, внутри квадратных скобок [] вместо индекса элемента используем чистую конструкцию `jq select(условие)`. В примере ниже мы ищем основной адрес, у которого стоит флаг `Primary: true`:

EmployeeInfo.Addresses[select(.Primary==true)].Country.Name -> primaryAddress.country.name

Addresses[select(.Type == "Main")].Street -> mainAddress.street

Чистый jq

В случае, если в источнике необходимо использовать какую-либо формулу, то в круглых скобках пишется чистое `jq` выражение:

(if .citizenship == "1" then 20 else null end) -> document.type.docTypeID

(!.Removed) -> active

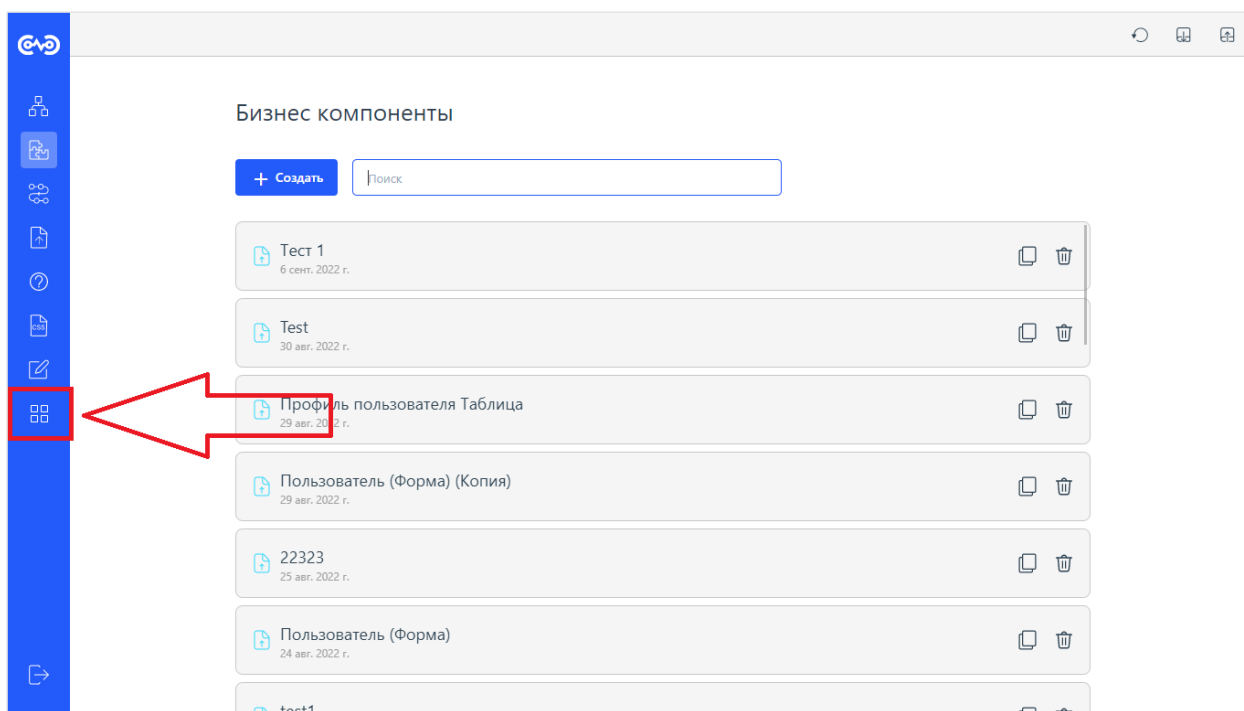
Бизнес-приложение

Бизнес-приложение — это многофункциональный программный комплекс, предназначенный для автоматизации ключевых бизнес-функций и процессов внутри компании.

Генерируемое **Gravitonum** бизнес приложение - это совокупность разработанной модели данных и компонентов экранных форм, привязанных к модели данных, которые связаны между собой бизнес процессами или определенной логикой переходов.

Как запустить бизнес-приложение?

Запустить бизнес-приложение можно через кнопку в тул-баре в интерфейсе платформы.



Как настроить бизнес-приложение?

Бизнес-приложение (далее приложение) имеет собственный механизм настройки пунктов меню, тулбара, инонок итд. После запуска и аутентификации, необходимо выбрать в приложении пункт **“Приложение”**, в левой части экрана в разделе **“Настройки”**, после чего откроется конфигурация приложения.

The image shows two screenshots of a web application interface. The top screenshot displays the 'Роли' (Roles) management page. The left sidebar contains navigation options under 'СПРАВОЧНИКИ' (Reference) and 'НАСТРОЙКИ' (Settings). The 'Роли' section is highlighted in the settings menu. The main content area shows a table of roles with columns for 'НАЗВАНИЕ' (Name), 'ОПИСАНИЕ' (Description), and 'ДЕЙСТВИЯ' (Actions). A red arrow points to the 'manager' role in the table. The bottom screenshot shows the 'Взять в работу' (Take to work) button and a code editor displaying a JSON configuration for the application menu. The configuration includes settings for logo, title, routes, and roles.

Роли

НАЗВАНИЕ	ОПИСАНИЕ	ДЕЙСТВИЯ
office-manager	Офис менеджер	
Accountant	Бухгалтер	
default-roles-docflow	\$(role_default-roles)	
director	Руководитель (ключевой согласователь)	
lawyer	Юрист	
manager	Менеджер	
hr	Отдел кадров	
platform-admin		
business-admin		
supervisor	супервизор, административная роль	

Взять в работу

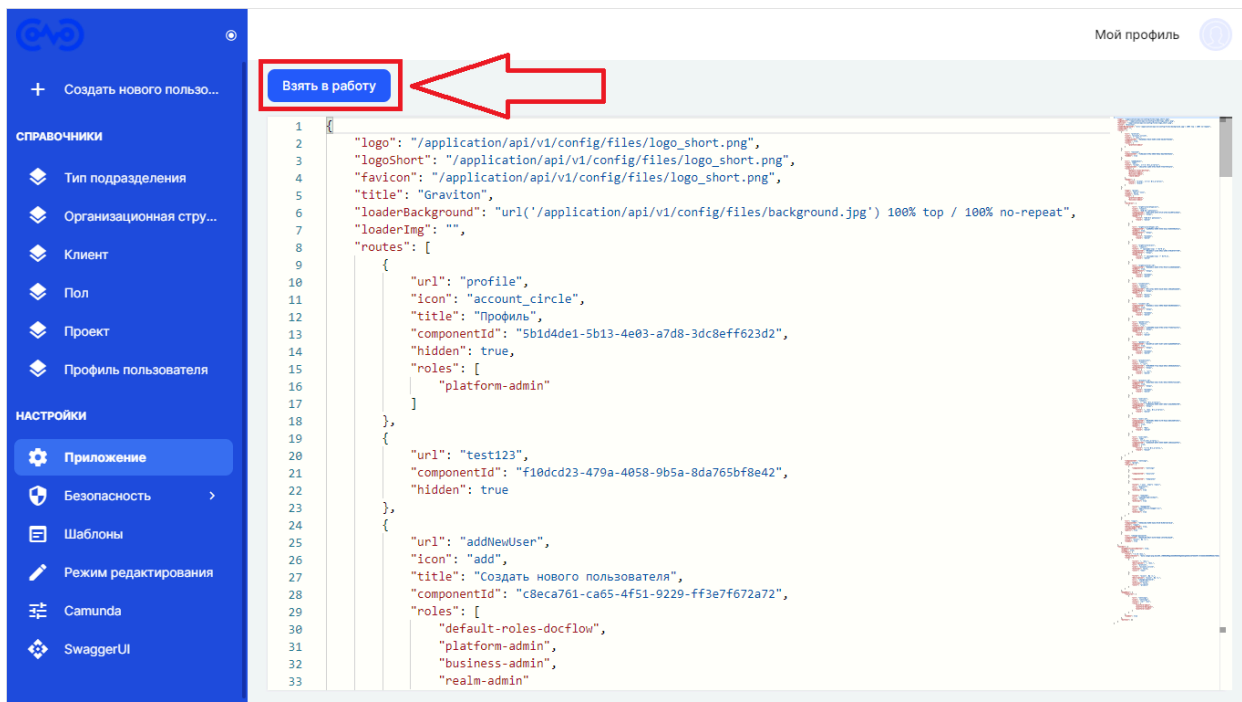
```

1  {
2    "logo": "/application/api/v1/config/files/logo_short.png",
3    "logoShort": "/application/api/v1/config/files/logo_short.png",
4    "favicon": "/application/api/v1/config/files/logo_short.png",
5    "title": "Graviton",
6    "loaderBackground": "url('/application/api/v1/config/files/background.jpg') 100% top / 100% no-repeat",
7    "loaderImg": "",
8    "routes": [
9      {
10       "url": "profile",
11       "icon": "account_circle",
12       "title": "Профиль",
13       "componentId": "5b1d4de1-5b13-4e03-a7d8-3dc8eff623d2",
14       "hidden": true,
15       "roles": [
16         "platform-admin"
17       ]
18     },
19     {
20       "url": "test123",
21       "componentId": "f18dcd23-479a-4058-9b5a-8da765bf8e42",
22       "hidden": true
23     },
24     {
25       "url": "addNewUser",
26       "icon": "add",
27       "title": "Создать нового пользователя",
28       "componentId": "c8eca761-ca65-4f51-9229-ff3e7f672a72",
29       "roles": [
30         "default-roles-docflow",
31         "platform-admin",
32         "business-admin",
33         "realm-admin"

```

Конфигурация меню

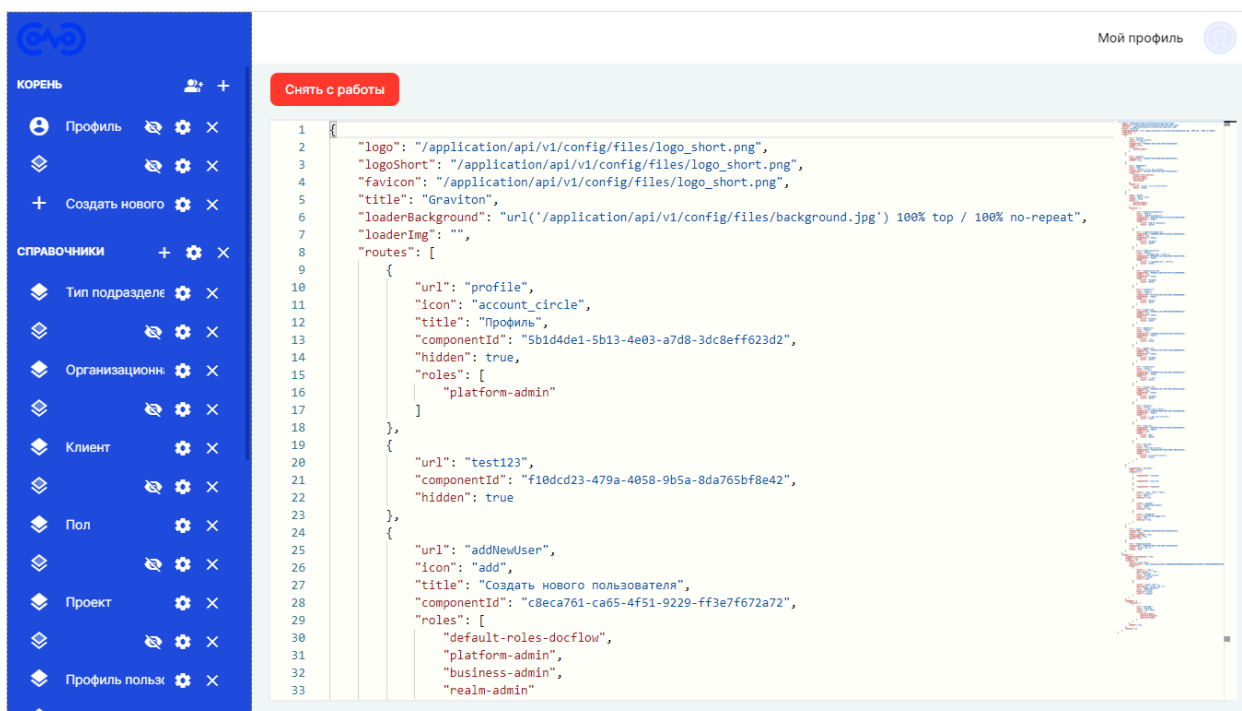
Конфигурацию разделов и пунктов меню можно выполнить с помощью специального интерфейса либо задать вручную. Для начала работы с конфигурацией необходимо нажать кнопку **“Взять в работу”**



При нажатии на кнопку **“Взять в работу”** интерфейс изменяется и слева появляется меню с настройками и всеми пунктами меню, которые в нем присутствуют. После нажатие кнопки, права на редактирование конфигурации переходят к текущему пользователю. Остальные пользователи при попытке нажать на кнопку видят сообщение, что конфигурация уже редактируется.

Конфигурация вручную

Конфигурация вручную проходит в редакторе приложения с редактирования настроек приложения в формате JSON. Ниже приведен код, который используется в создании конфигурации.



Заголовок

```
{
  favicon: string; // Иконка сайта на закладке браузера
```



```

title: string; // Заголовок сайта на закладке браузера
logo: string; // Логотип на левой панели меню в развёрнутом меню навигации
logoShort: string; // Логотип на левой панели меню в свёрнутом режиме
notificationPollingIntervalMS : number; // Интервал получения уведомлений. По умолчанию
равен 30000 (30 сек)
toolbar: Toolbar; // Настройки верхней панели
routes: Route[]; // Настройки навигации и существующих страниц
}

```

Тулбар

```

{
  userMenu : { // Меню с правой стороны toolbar'a
    icon: string; // Иконка
    color : string; // Цвет иконки
    title: string; // Заголовок
    description: string; // Описание под заголовком
    url: string; // Ссылка (поддерживает переменные вида ${var})
    external: boolean; // Открывать ссылку в новой вкладке
    roles: string[]; // Роли, которым будет доступен этот пункт
  }[];
  buttons: { // Кнопки на верхней панели
    title: string; // Надпись на кнопке
    url: string; // Ссылка (поддерживает переменные вида ${var})
    external: boolean; // Открывать ссылку в новой вкладке
    color: 'primary' | 'accent' | 'warn'; // Цвет из стандартного набора цветов
    roles: string[]; // Роли, которым будет доступна эта кнопка
  }[];
}

```

Группа

```

{
  type: 'group'; // Тип route
  title: string; // Заголовок группы
  hidden: boolean; // Скрывает route в навигационном меню
  children: Route[]; // Дочерние элементы (route)
  roles: string[]; // Роли которым разрешена эта группа
  inheritRoles: boolean; // Наследовать роли от родителя (по умолчанию true)
}

```

Пример конфигурации

```

{
  "title": "Вакансии",
  "type": "group",
  "children": [
    {
      "url": "vacancyall",
      "title": "Все",
      "icon": "recent_actors",
      "componentId": "485550b8-e39b-454f-abc2-b23ee8459580",
      "children": [
        {
          "url": "employee",
          "title": "Сотрудник",
          "componentId": "e2458aaa-1a0c-4694-aa6f-59ac29b8fe17"
        }
      ]
    }
  ]
}

```

```

    }
  ]
},
{
  "url": "vacancyopen",
  "title": "Открытые",
  "componentId": "a7a58f31-f5d0-4ace-a2fc-c99645ae98c7",
},
{
  "url": "vacancyclose",
  "title": "Закрытые",
  "componentId": "9995b09b-67e5-453a-b761-7adf8eff7a30"
}
]
}

```

Группа с дочерней группой

```

{
  url: string;           // URL поддерживает переменные в виде '/user/:var/' - где ':var:' это
                          // переменная. Не поддерживает query параметры
  icon: string;         // Иконка в навигационном меню
  title: string;        // Название в навигационном меню
  hidden: boolean;     // Скрывает route в навигационном меню
  children: Route[];   // Дочернии route
  roles: string[];     // Роли которым разрешен этот route
  inheritRoles: boolean; // Наследовать роли от родителя (по умолчанию true)
  chip: {              // Чип отображается справа от элемента навигации
    color: string;     // Цвет чипа
    url: string;       // Путь для graphql запроса
    graphql: string;   // Строка graphql запроса, получение данных для чипа (поддержи
                          // вает переменные вида ${var})
    variables: Params; // Переменные для graphql запроса (поддерживает переменные
                          // вида ${var})
  }
}

```

Пример конфигурации

```

{
  "title": "СПРАВОЧНИКИ",
  "url": "spra",
  "roles": ["admin"],
  "children": [
    {
      "url": "employee",
      "title": "Сотрудник",
      "componentId": "e2458aaa-1a0c-4694-aa6f-59ac29b8fe17"
    },
    {
      "url": "spec",
      "title": "Специализация",
      "componentId": "7da46d04-a15c-41a7-a5c9-1950ee563f78"
    },
    {
      "url": "businessUnit",

```

```

    "title": "Бизнес-Юнит",
    "componentId": "21aa0b1e-a820-4f2a-95ea-82a36fed19f0"
  }
]
}

```

Обычная ссылка

```

{
  url: string; // URL, поддерживает переменные в виде '/user/:var:/' - где ':var:'
переменная. Не поддерживает query параметры
  external: boolean; // Открывать ссылку в новой вкладке
  icon: string; // Иконка в навигационном меню
  title: string; // Название в навигационном меню
  header: { // Шапка страницы
    title: string, // Заголовок в шапке
    style?: 'black' | 'white' // Цвет заголовка в шапке (по умолчанию black)
    tooltip?: string // Тултип рядом с заголовком страницы
  };
  queryParams?: Params; // Query параметры
  componentId?: string; // ID компонента
  heightPanel?: string; // Размеры синей плашки
  isIndexPage?: boolean; // Помечает страницу как главную
  hidden?: boolean; // Скрывает route в навигационном меню
  roles?: string[]; // Роли, которым разрешен этот route
  inheritRoles?: boolean; // Наследовать роли от родителя (по умолчанию true)
  chip?: { // Чип отображается справа от элемента навигации
    color: string; // Цвет чипа
    url: string; // Путь для graphql запроса
    graphql: string; // Строка graphql запроса, получение данных для чипа
    variables: Params; // Переменные для graphql запроса
  },
  breadcrumbs?: { // Массив хлебных крошек
    title: string; // Заголовок крошки. Поддерживаются query переменные. В заго
ловке выглядят как ${name}
    url: string; // Ссылка куда крошка ведёт
  }[];
}

```

Пример конфигурации

```

{
  "url": "vacancyopen",
  "title": "Открытые",
  "componentId": "a7a58f31-f5d0-4ace-a2fc-c99645ae98c7",
  "chip": {
    "color": "warn",
    "url": "/security",
    "graphql": "query($realm: String!, $value: String){userCount(realm: $realm, value: $value)}"
  },
  "variables": {
    "value": "aimc.io"
  }
},
"heightPanel": "200px",
"header": {

```

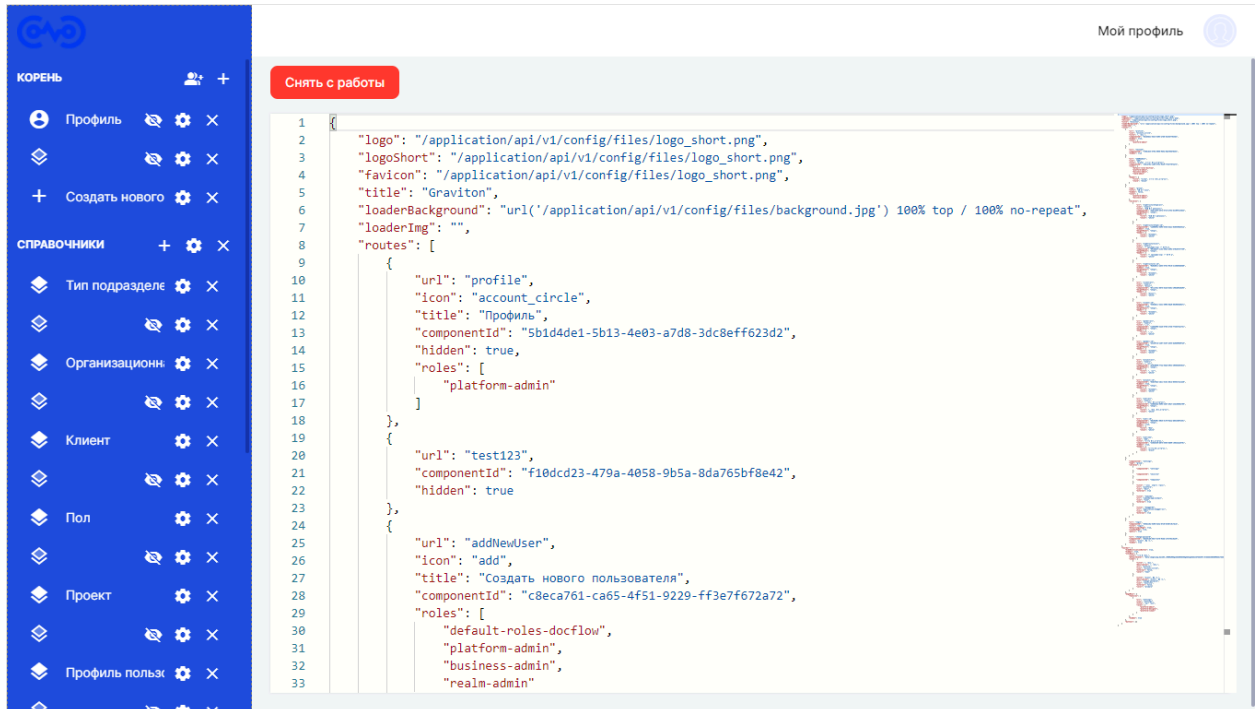
```

    "title": "Важный заголовок для страницы",
    "style": "white"
  },
  "breadcrumbs": [
    {"title": "Главная", "url": "roles"},
    {"title": "Справочная", "url": "users"}
  ]
}

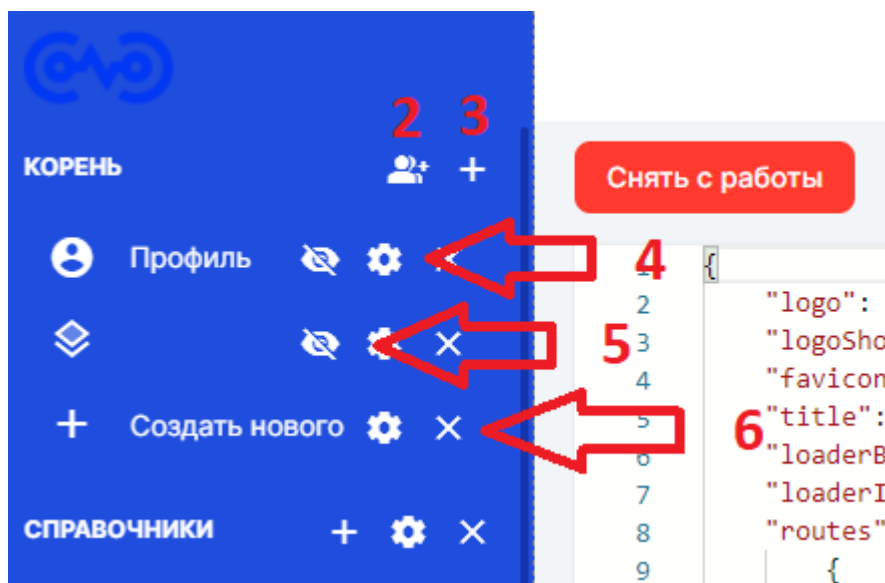
```

Конфигурация с помощью приложения

Конфигурация с помощью приложения позволяет быстро создавать недостающие и необходимые странички, а так же настраивать их видимость.



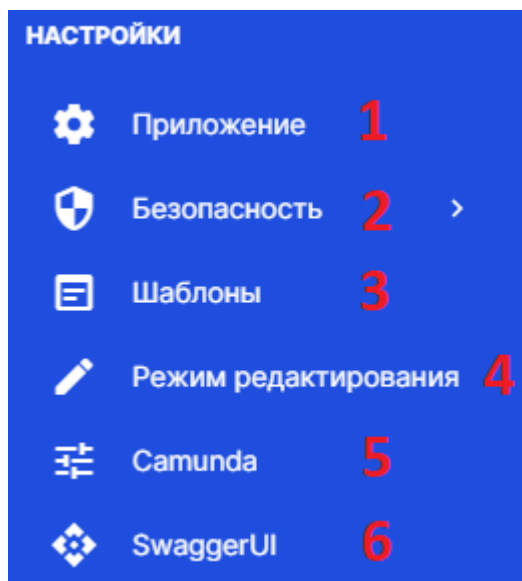
Разберемся с левой панелью меню:



1. **Корень** - по сути это тот же Route, который используется в конфигурации с помощью кода

2. **Группа** - позволяет добавлять группу, в которую в дальнейшем будут добавлены страницы
3. **Добавить** - позволяет добавить новый пункт меню
4. **Шестеренка** - позволяет настраивать пункт меню или группу
5. **Перечеркнутый глаз** - позволяет устанавливать видимость пункта меню, по необходимости
6. **Крестик** - позволяет удалить не нужный пункт

Так же в этой же панели есть те пункты, которые нельзя изменить через эту панель, они объединены в группу “**Настройки**”:



1. **Приложение** - конфигурация приложения
2. **Безопасность** - раздел безопасности с настройками пользователей, ролей и правил
3. **Шаблоны** - раздел для создания шаблонов
4. **Режим редактирования** - позволяет перейти в редактор **Модели данных, экранных форм, бизнес-процессов** (добавить ссылки)
5. **Camunda** - позволяет перейти в приложение Camunda Cockpit
6. **SwaggerUI** - позволяет перейти в документацию к REST API, где можно посмотреть созданные запросы

Настройки пункта меню

Чтобы открыть настройки пункта меню, необходимо нажать на “шестеренку”, после этого откроется диалоговое окно с настройками.

Пункт меню: Новый пункт ×

Заголовок пункта меню **1**
Новый пункт

URL **2**

Иконка **3**

Бизнес-компонент **4**

Роли, которым разрешен этот url **5**

Шапка страницы

Чип

Хлебные крошки

Скрыть в меню навигации

Сохранить

1. **Заголовок пункта меню** - позволяет установить название пункта меню в левой панели, автоматически называется “Новый пункт”, можно изменить
2. **URL** - ссылка, по которой будет вестить нажатие на пункт меню
3. **Иконка** - позволяет устанавливать иконку для пункта, можно использовать иконки Material Icons
4. **Бизнес-компонент** - позволяет выбрать компонент, который будет отображаться на странице
5. **Роли, которым разрешен этот url** - позволяет устанавливать ограничение на доступ к этой странице/пункту меню по роли

Пункт меню: Новый пункт ✕

Шапка страницы ^

Заголовок страницы * **1**

Стиль заголовка
black **2** ▼

Высота верхней синей плашки **3**

Тултип **4**

Чип ▼

Хлебные крошки ▼

Скрыть в меню навигации

Сохранить

1. **Заголовок страницы** - с помощью него устанавливается заголовок страницы, который будет на ней отображаться
2. **Стиль заголовка** - можно установить стиль заголовка, черный или белый он будет
3. **Высота верхней синей плашки** - позволяет установить высоту плашки
4. **Тултип** - позволяет установить описание вкладки

Пункт меню: Новый пункт ×

Путь * **1**

Цвет чипа **2**

GraphQL запрос **3**

GraphQL переменные **4**

Пулинг интервал (ms) **5**
15000

Сохранить

1. **Путь запроса** - путь запроса указывает куда будет направлен запрос к бэкенду или к приложению
2. **Цвет чипа** - позволяет указать цвет чипа, который буде выводится рядом с пунктом меню
3. **GraphQL запрос** - позволяет написать graphql запрос к бэкенду или к приложению
4. **GraphQL переменные** - позволяет указать переменные, которые будут использованы в запросе
5. **Пулинг интервал (ms)** - позволяет указывать интервал с какой периодичностью будет отправляться запрос, указывается в миллисекундах

Пункт меню: Новый пункт ✕

Чип ▾

Хлебные крошки ▴

Заголовок * **1** URL **2** ✕

[+ Добавить](#)

Скрыть в меню навигации **3**

Открывать в новой вкладке **4**

Главная страница в текущем родителе **5**

Наследовать роли от родителя **6**

Контейнер на всю ширину **7**

Общедоступная страница **8**

[Сохранить](#)

1. **Заголовок** - позволяет устанавливать заголовок для навигации
2. **URI** - позволяет устанавливать url, куда будет вести навигация
3. **Скрыть в меню навигации** - позволяет устанавливать отображение пункта в меню
4. **Открывать в новой вкладке** - при активированном чек боксе позволяет открывать страницу в новой вкладке
5. **Главная страница в текущем родителе** - позволяет установить страницу главной в группе
6. **Наследовать роли от родителя** - позволяет пункту наследовать роли от родителя, группы
7. **Контейнер на всю ширину** - позволяет растянуть страницу на всю ширину
8. **Общедоступная страница** - позволяет указать, что данная страница общедоступна

Шаблоны / Отчеты

Шаблон состоит из следующих блоков:

1. Путь запроса
2. Запрос
3. Заголовок
4. Сообщение

“Запрос” полностью поддерживает graphql синтаксис.

Блок “Сообщения” и “Прикрепленный файл” имеют один формат для работы с переменными.

















Пример для получения элемента сущности:

Путь запроса
application

```
query {  
  cityFindById(id: "513c27b3-3745-4838-a11f-3f6d44853ba6") {  
    id  
    name  
    code  
  }  
}
```

Сообщение

Заголовок
Город

B I U    H1 H2   x₂ x²    Normal  
Sans Serif      

Текущий город: {cityFindById.name}

Пример для получения списка:

Запрос

Путь запроса
application

```
query cityFind {cityFind {  
  id  
  code  
  name  
}}
```

Сообщение

Заголовок
Список городов

B *I* U   **H1** **H2**   x_2 x^2    Normal  
Sans Serif     

Список городов
{#each cityFind}

Город: {it.name} {it.code}

{/each}

Безопасность

Пользователи

Пользователь – субъект, имеющий Учётную запись и использующий Приложение в своей работе

Учетная запись - информация, необходимая для прохождения Пользователем процессов аутентификации и авторизации в Приложении

Управление Пользователями доступно Администраторам в разделе **Безопасность** → **Пользователи**

ДАТА СОЗДАНИЯ	ЛОГИН	ИМЯ	ФАМИЛИЯ	EMAIL	РОЛИ	ДЕЙСТВИЯ
10 авг. 2022 г.	aantropova	Анна	Антропова	aantropova@ai...	default-roles-d...	
4 авг. 2022 г.	administrator	LCP	Administrator	admin@keycloa...	default-roles-d...	
4 авг. 2022 г.	aformin	Artem	Formin	aformin@gravit...	default-roles-d...	
4 авг. 2022 г.	akartintseva	Anastasiya	Kartintseva	akartintseva@g...	default-roles-d...	
4 авг. 2022 г.	akharintsev	Andrey	Kharintsev	akharintsev@gr...	default-roles-d...	
4 авг. 2022 г.	ashchurin	Artem	Shchurin	ashchurin@gra...	default-roles-d...	
4 авг. 2022 г.	dyartsev	Dmitry	Yartsev	dyartsev@gravi...	default-roles-d...	
4 авг. 2022 г.	episarev	Evgeniy	Pisarev	episarev@gravi...	default-roles-d...	
4 авг. 2022 г.	imoiseev	Ivan	Moiseev	imoiseev@grav...	default-roles-d...	
4 авг. 2022 г.	pkalinin	Pavel	Kalinin	pkalinin@gravit...	default-roles-d...	
4 авг. 2022 г.	rozhek	Dmitry	Rozhek	rozhek@gravi...	default-roles-d...	

Пользователи

Возможности:

- Добавление Пользователя
- Просмотр и редактирования данных Пользователя
- Управление ролями Пользователя
- Блокирование учетной записи Пользователя
- Сброс пароля учетной записи Пользователя

Роли

Роль — совокупность возможностей, которыми обладает Пользователь

Управление Ролями доступно Администраторам в разделе **Безопасность** → **Роли**

Мой профиль

Создать нового пользо...

СПРАВОЧНИКИ

- Тип подразделения
- Организационная стру...
- Клиент
- Пол
- Проект
- Профиль пользователя

НАСТРОЙКИ

- Приложение
- Безопасность**
 - Пользователи
 - Роли**
 - Правила
- Шаблоны
- Режим редактирования

Роли

Поиск

+ Создать новую роль

НАЗВАНИЕ :	ОПИСАНИЕ :	ДЕЙСТВИЯ
office-manager	Офис менеджер	
Accountant	Бухгалтер	
default-roles-docflow	\$(role_default-roles)	
director	Руководитель (ключевой согласователь)	
lawyer	Юрист	
manager	Менеджер	
hr	Отдел кадров	
platform-admin		
business-admin		
supervisor	супервизор, административная роль	

Роли

Возможности:

- Добавление Роли
- Просмотр и редактирование Роли
- Удаления Роли

Данные пользователей и настройки ролей могут храниться как непосредственно на сервере Keusloak, так и в интегрированных с ним службах каталогов LDAP или Active Directory

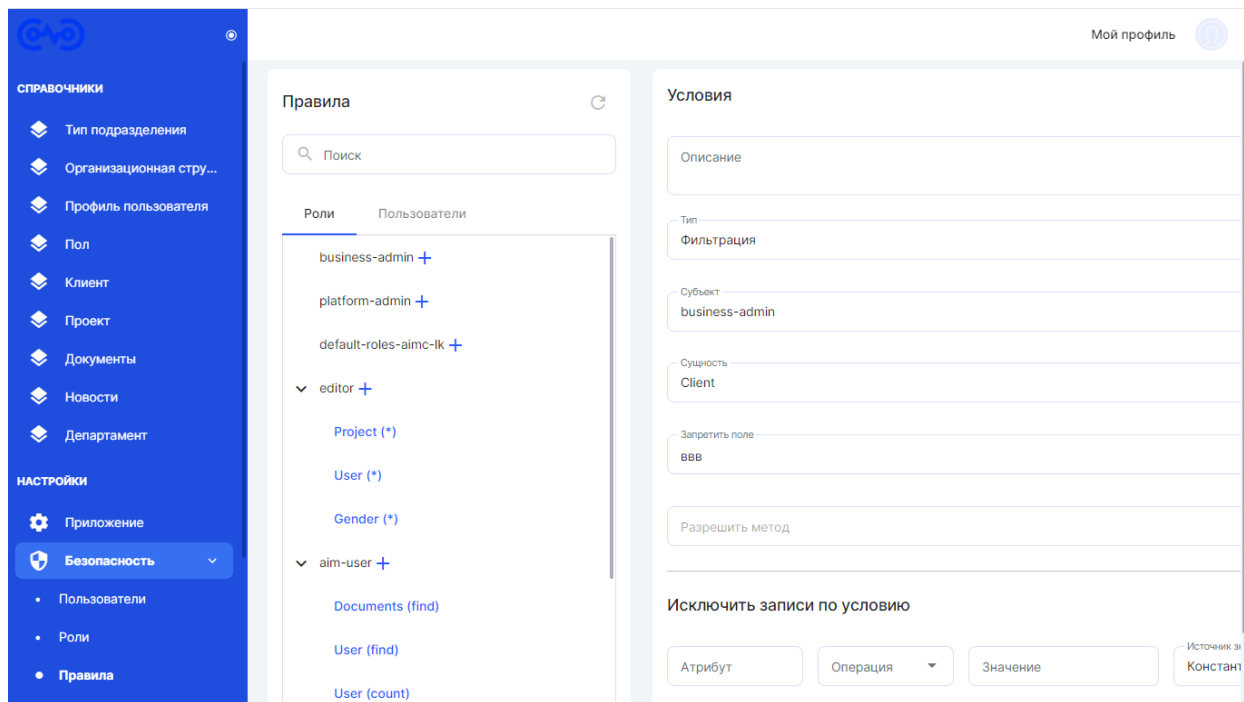
Правила

Платформа позволяет настраивать возможности работы с Приложением субъекту (*Роли* или *Пользователю*) с использованием правил разграничения доступа на основе атрибутов.

Разграничение доступа на основе атрибутов (англ. *Attribute-Based Access Control, ABAC*) — модель контроля доступа к объектам, основанная на анализе правил для атрибутов объектов или субъектов, возможных операций с ними и окружения, соответствующего запросу.

Атрибуты могут быть простыми или комплексными (вложенными): firstName, project.name, client.phones.main.number

Управление Правилами доступно Администраторам в разделе **Безопасность** → **Правила**



Пользователи

Возможности:

- Добавление Правила
- Просмотр и редактирование Правила
- Удаление Правила

Интеграция

Сервисы нотификаций

Платформа поддерживает несколько видов уведомлений из коробки, таких как:

- email,
- telegram,
- sms (интеграция со шлюзом Заказчика).

Интерфейс платформы позволяет гибко настраивать шаблон уведомлений.

Импорт данных из Excel

Импорт данных из эксель используется, чтобы заполнять справочники базы данных. Чтобы выполнить запрос, необходимо иметь программу, с помощью которой можно работать с разными типами запросов: rest, GraphQL и другими.

Что требуется для импорта данных?

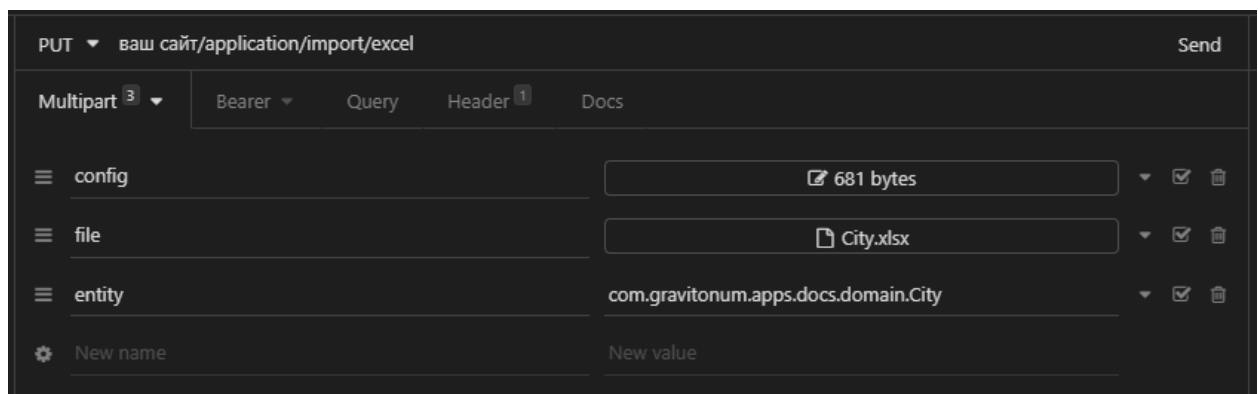
1. Insomnia или другая внешняя программа для отправки запросов
2. Токен для запросов (можно настроить запросом и после в запросе его использовать)
3. Ссылка на приложение/сайт, куда необходимо загрузить данные из excel

4. Файл excel, в котором есть необходимые данные, которые и надо загрузить
5. Знать полное название сущности, в которую будут загружены данные.

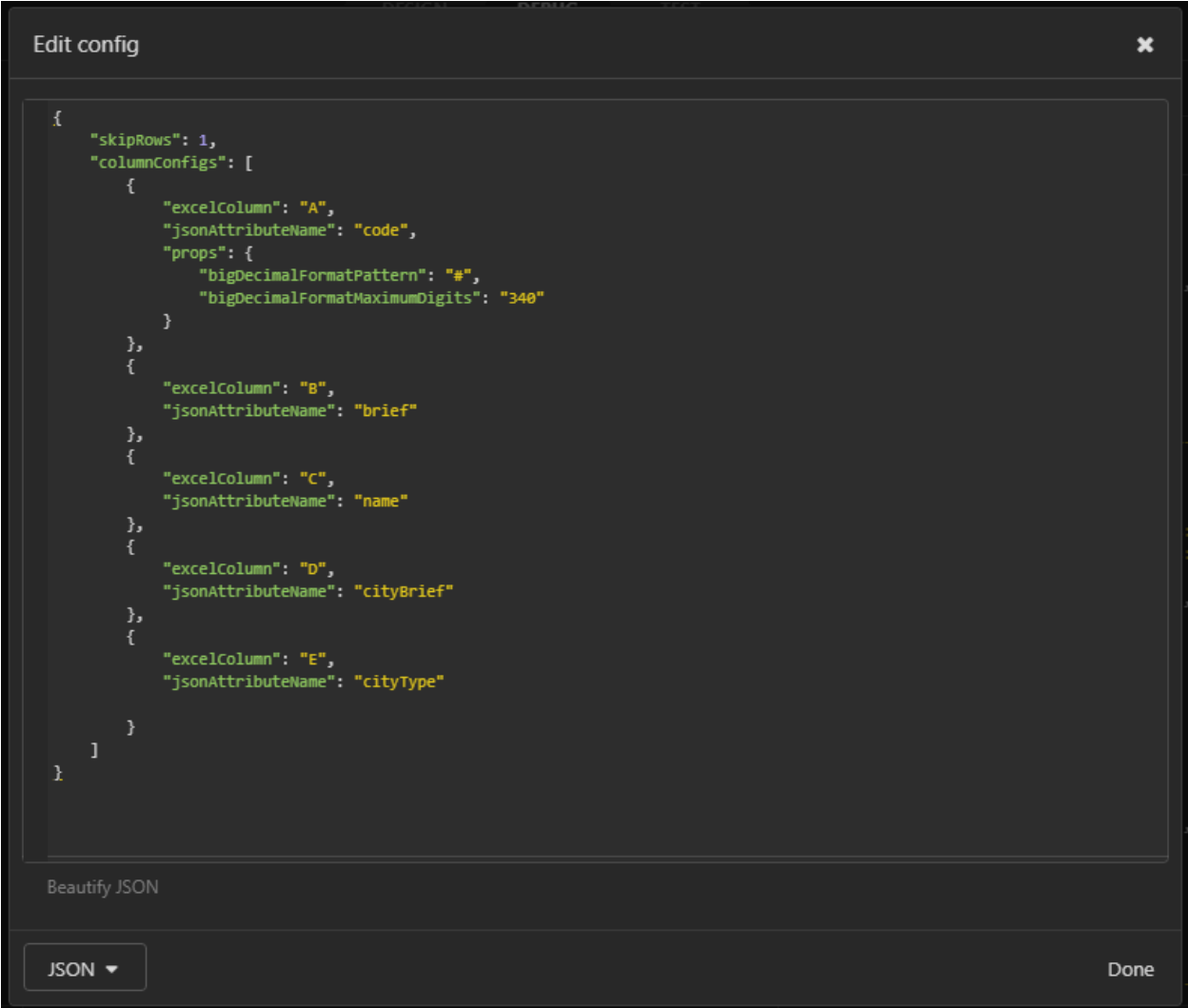
Настройка запроса

1. Создать новый запрос с методом PUT и телом Multipart Form
2. Настроить аутентификатор на получение токена, обычно используется настройка Bearer Token с переменной Response Body Attribute
3. В URL запроса необходимо установить ссылку на "ваш сайт"/application/import/excel
4. В теле запроса установить 3 параметра
 1. config - содежит json конфигурацию, см пример ниже > Для конфигурации выбирается Text (Multi-line), т.к. в нем будет указана json настройка. > Обязательно необходимо выбрать формат конфига JSON. Это можно сделать внизу слева окна настройки конфигурации.
 2. file - Excel файл с данными, который надо импортировать
 3. entity - полное имя сущности, в которую будут записываться данные из приложенного Excel файла > В пункте 4.2. в типе переменной выбирается и загружается Файл

Префикс перед наименованием сущности - это пакет, в котором находятся все сущности. Его можно найти в [Модели данных](#), выбрав интересующую сущность и посмотрев в [Настройках сущности](#) пункт "Модуль". Данное поле является не редактируемым, но информацию из него можно скопировать.



Пример конфигурации



The screenshot shows a dark-themed 'Edit config' window with a close button in the top right. The main area contains a JSON configuration for column settings. At the bottom, there is a 'Beautify JSON' button and a dropdown menu currently set to 'JSON', with a 'Done' button on the far right.

```
{
  "skipRows": 1,
  "columnConfigs": [
    {
      "excelColumn": "A",
      "jsonAttributeName": "code",
      "props": {
        "bigDecimalFormatPattern": "#",
        "bigDecimalFormatMaximumDigits": "340"
      }
    },
    {
      "excelColumn": "B",
      "jsonAttributeName": "brief"
    },
    {
      "excelColumn": "C",
      "jsonAttributeName": "name"
    },
    {
      "excelColumn": "D",
      "jsonAttributeName": "cityBrief"
    },
    {
      "excelColumn": "E",
      "jsonAttributeName": "cityType"
    }
  ]
}
```

```
{
  "skipRows": 1,
  "columnConfigs": [
    {
      "excelColumn": "A",
      "jsonAttributeName": "code",
      "props": {
        "bigDecimalFormatPattern": "#",
        "bigDecimalFormatMaximumDigits": "340"
      }
    },
    {
      "excelColumn": "B",
      "jsonAttributeName": "brief"
    },
    {
      "excelColumn": "C",
      "jsonAttributeName": "name"
    },
    {
      "excelColumn": "D",
      "jsonAttributeName": "cityBrief"
    }
  ]
}
```



```

    },
    {
      "excelColumn": "E",
      "jsonAttributeName": "cityType"
    }
  ]
}

```

"skipRows": 1, - если в файле excel первая строка отведена под заголовки столбцов, то её необходимо пропустить, чтобы импорт прошел правильно. Указывается количество строк, которые надо пропустить методу, чтобы считать данные.

"excelColumn": "A" - выбирается колонка из которой будут браться данные.

"jsonAttributeName": "code" - указывается атрибут в сущности в Модели данных, куда будут загружаться данные из колонки excel.

"props": {"bigDecimalFormatPattern": "#", "bigDecimalFormatMaximumDigits": "340"} - используется, чтобы конвертировать цифры и числа в тип String для базы данных.

Так же данная функциональность работает и со вложенными полями, например cityType.code, cityType.name. То есть если у вас есть две сущности/справочника, которые имеют связь, то можно беспрепятственно загрузить оба справочника одним файлом.

```

{
  "skipRows": 1,
  "columnConfigs": [
    {
      "excelColumn": "A",
      "jsonAttributeName": "code",
      "props": {
        "bigDecimalFormatPattern": "#",
        "bigDecimalFormatMaximumDigits": "340"
      }
    },
    {
      "excelColumn": "B",
      "jsonAttributeName": "brief"
    },
    {
      "excelColumn": "C",
      "jsonAttributeName": "name"
    },
    {
      "excelColumn": "D",
      "jsonAttributeName": "cityBrief"
    },
    {
      "excelColumn": "E",
      "jsonAttributeName": "cityType.name"
    },
    {
      "excelColumn": "F",
      "jsonAttributeName": "cityType.code",

```

```

    "props": {
      "bigDecimalFormatPattern": "#",
      "bigDecimalFormatMaximumDigits": "340"
    }
  ]
}

```

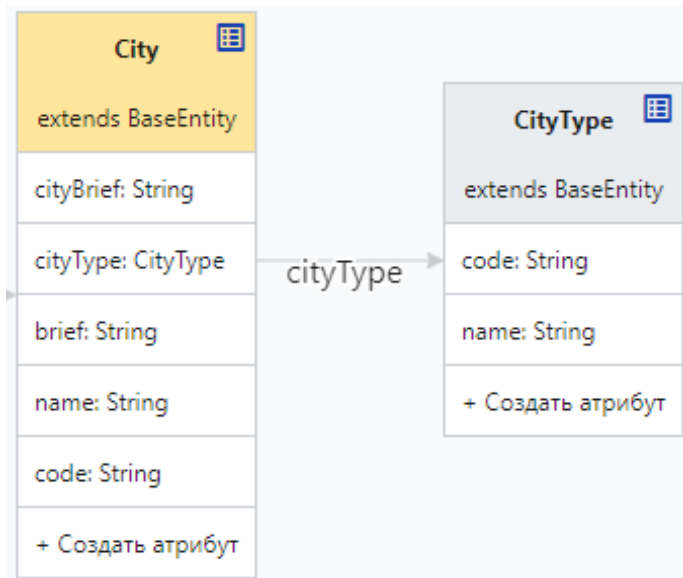
Пример excel файла

	A	B	C	D	E
1	Код	Сокращенное название	Название	Сокращенный тип	Тип
2	10001092564	АБАКАН	АБАКАН	нп	Населенный пункт
3	10001090676	ALDAN	ALDAN	нп	Населенный пункт
4	10003780110	ALEKSANDROVSK	ALEKSANDROVSK	нп	Населенный пункт
5	10001100964	ALEXANDROV	ALEXANDROV	нп	Населенный пункт
6	10001090712	ALMETYEVSK	ALMETYEVSK	нп	Населенный пункт
7	10001102004	ANADYR	ANADYR	нп	Населенный пункт
8	10001091600	APASTOVO	APASTOVO	нп	Населенный пункт
9	10001091755	ARKHANGELSK	ARKHANGELSK	нп	Населенный пункт
10	10001091769	ARMAVIR	ARMAVIR	нп	Населенный пункт
11	10001091601	ARSK	ARSK	нп	Населенный пункт
12	10001096388	ARZAMAS	ARZAMAS	нп	Населенный пункт
13	10001096177	ASHA	ASHA	нп	Населенный пункт
14	10003780111	ASINO	ASINO	нп	Населенный пункт
15	10001090063	ASTRAKHAN	ASTRAKHAN	нп	Населенный пункт
16	10001091602	AZNAKAEVO	AZNAKAEVO	нп	Населенный пункт
17	10001098560	BAIKONUR	BAIKONUR	нп	Населенный пункт
18	10001093867	BALAKOVO	BALAKOVO	нп	Населенный пункт
19	10001102025	BALASHIKHA	BALASHIKHA	нп	Населенный пункт
20	10001094846	BALASHOV	BALASHOV	нп	Населенный пункт

City
extends BaseEntity
brief: String
cityType: String
code: String
name: String
cityBrief: String

В экселе и в модели данных атрибуты и столбцы отличаются названием, поэтому необходимо проанализировать какой столбец в какой атрибут будет сохраняться. * Код будет сохранен в code * Сокращенное название будет сохранено в brief * Название будет сохранено в name * Сокращенный тип будет сохранен в cityBrief * Тип будет сохранен в cityType

	A	B	C	D	E	F
1	Код	Сокращенное название	Название	Сокращенный тип	Тип	Код типа
2	10001092564	АБАКАН	АБАКАН	нп	Населенный пункт	1
3	10001090676	АЛДАН	АЛДАН	нп	Населенный пункт	1
4	10003780110	ALEKSANDROVSK	ALEKSANDROVSK	нп	Населенный пункт	1
5	10001100964	ALEXANDROV	ALEXANDROV	нп	Населенный пункт	1



Если загружается файл с конфигурацией и для вложенных полей, то будут заполняться сразу две сущности. Необходимо лишь добавить в файл, например уникальное значение для второго справочника. Согласно связям между двумя сущностями значения будут сохранены в те атрибуты, куда было указано. * Тип будет сохранен в cityType.name * Код типа будет сохранен в cityType.code

Рекомендации и параметры для конвертации

Рекомендуется установить поле, например, code уникальным, в настройках атрибута, чтобы можно было избежать задвоений при повторном импорте.

Параметры для конвертации в String

- Конвертация цифр и чисел

```
"props": {
  "bigDecimalFormatPattern": "#",
  "bigDecimalFormatMaximumDigits": "340"
}
```

- Конвертация даты

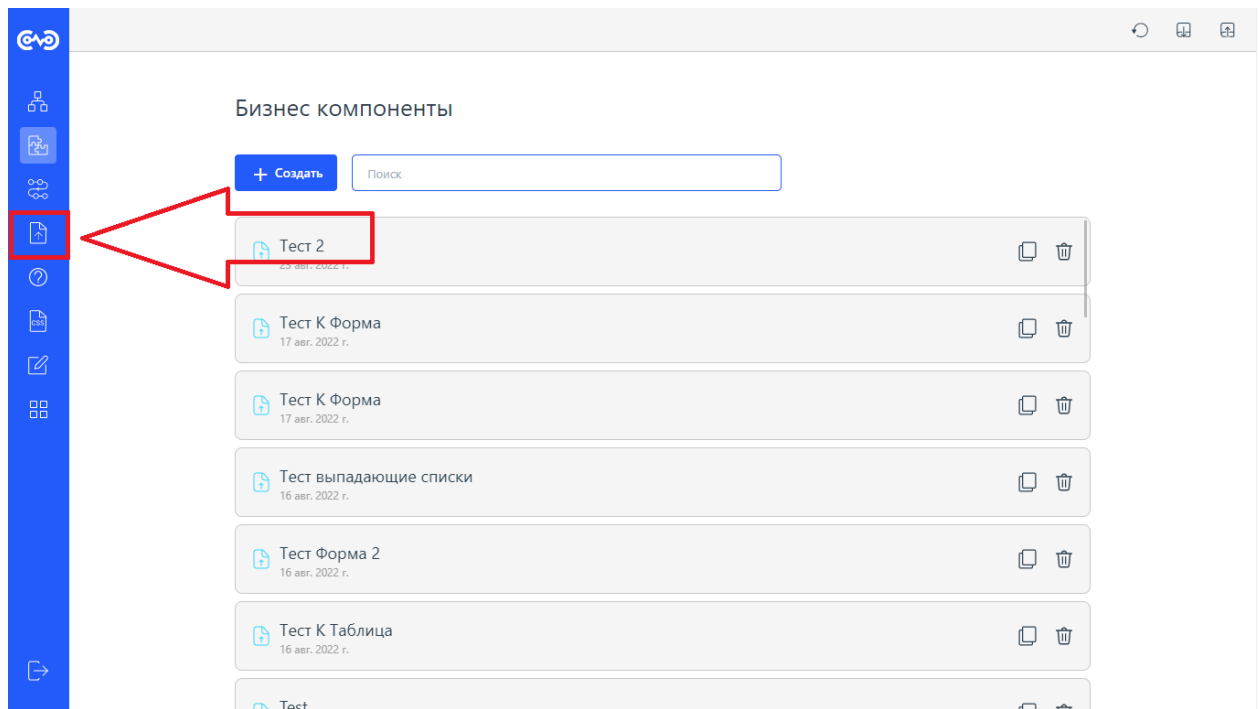
```
"props": {
  "dateFormat": "dd.MM.yyyy"
}
```

Хранилище ресурсов

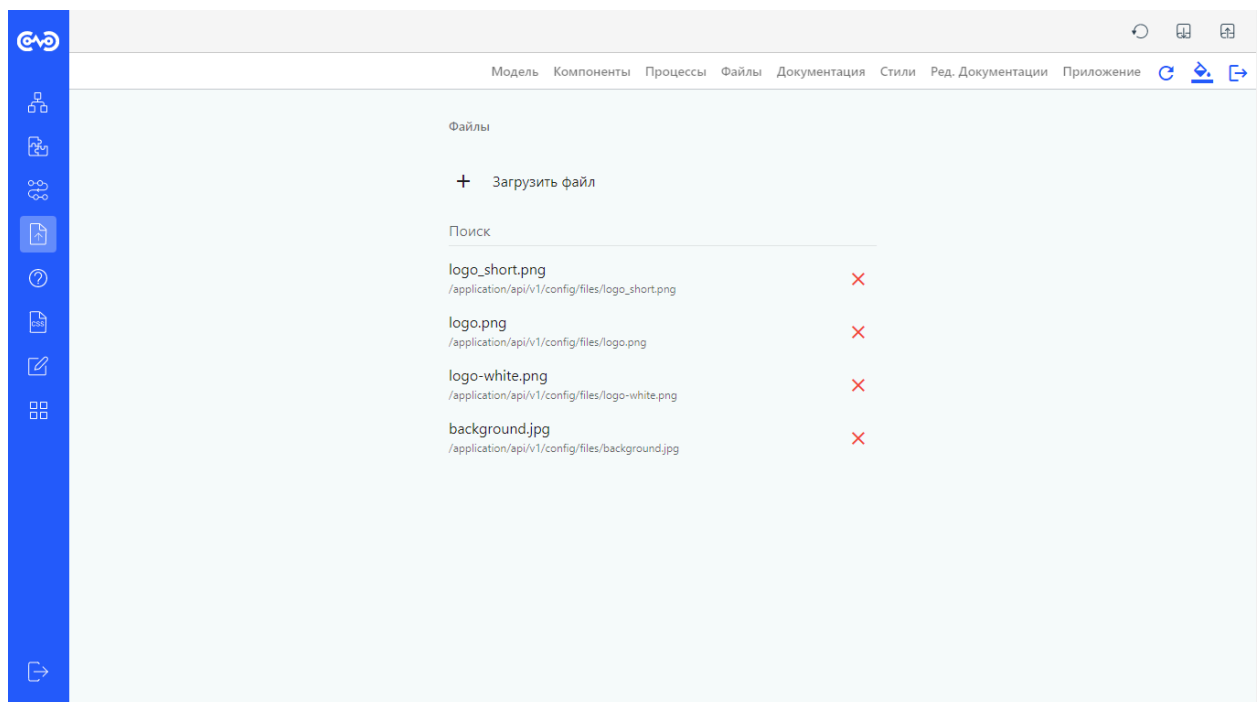
Хранилище файлов используется для хранения и использования файлов в дальнейшем, но до развертывания в Прод-режим. В таком режиме будет недоступна загрузка файлов.

Загрузка файлов

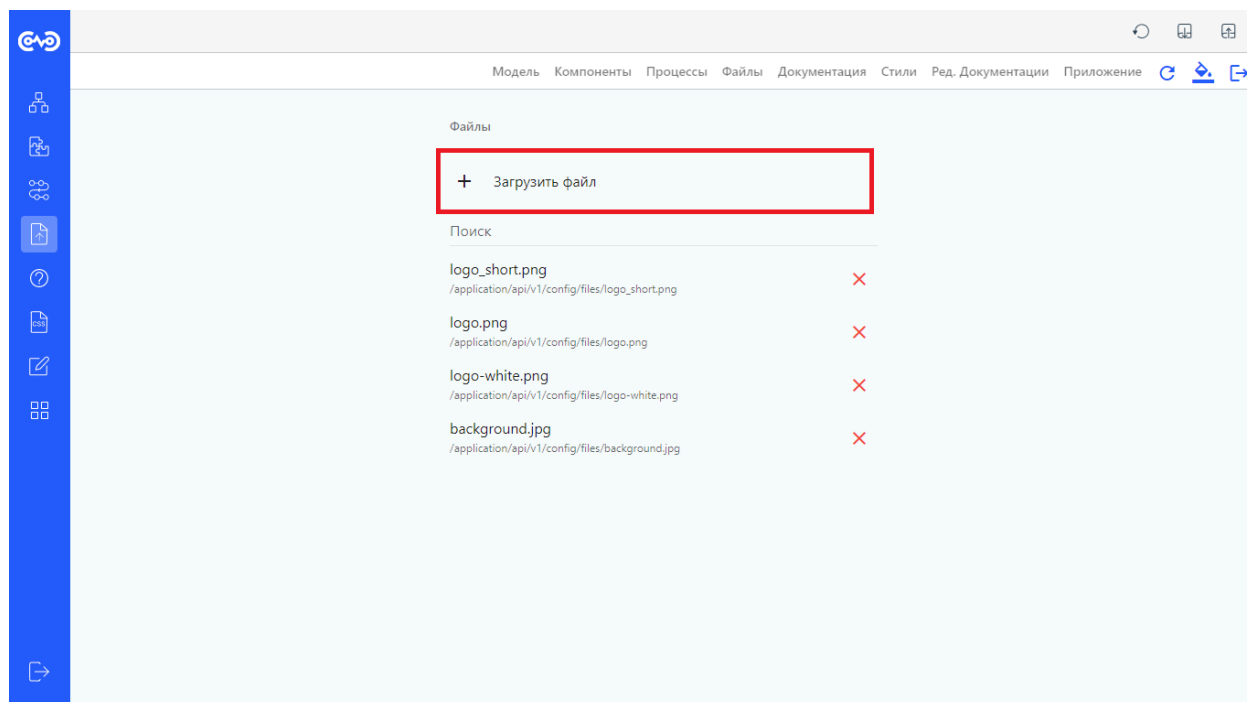
Для использования файла в платформе/приложении необходимо загрузить файл. Для этого необходимо перейти во вкладку “**Файлы**”, на неё можно перейти через тулбар, который располагается слева.



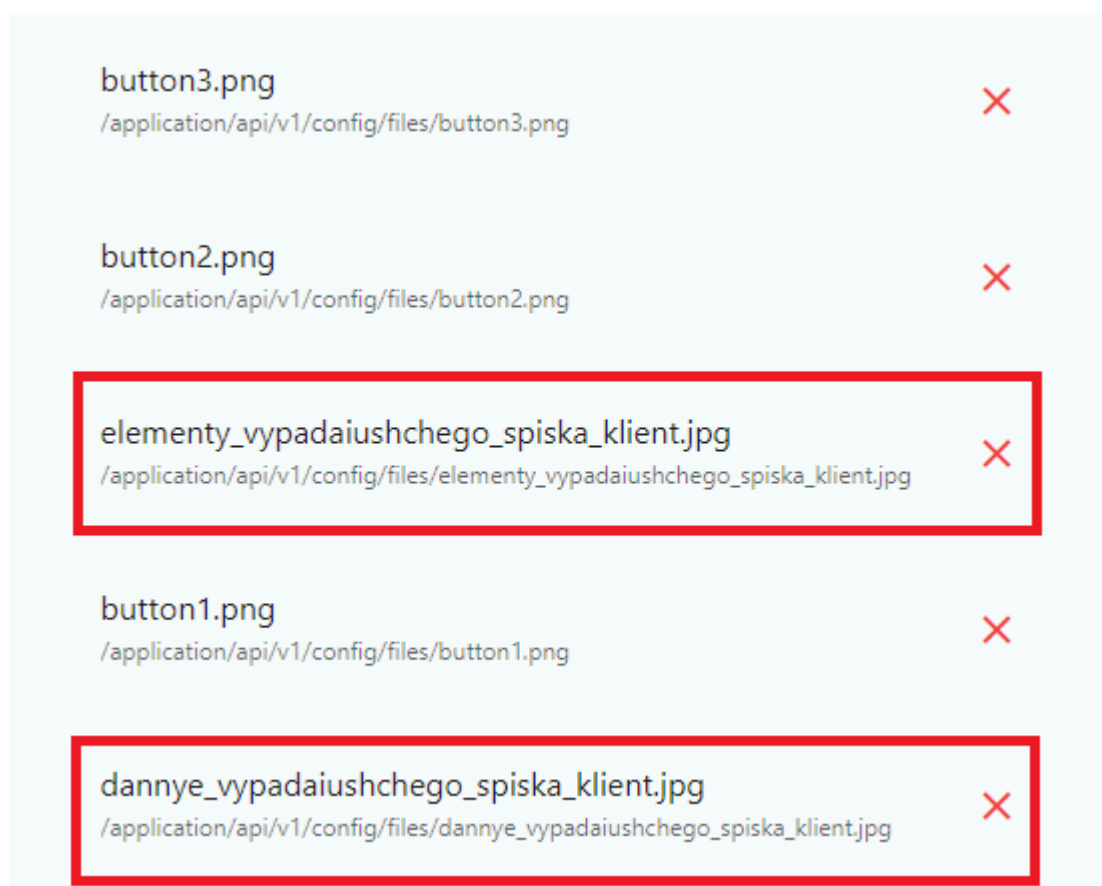
После перехода откроется экран с загрузкой файлов



Далее нажать на **“Загрузить файл”** и выбрать необходимый файл в открывшемся проводнике операционной системы.



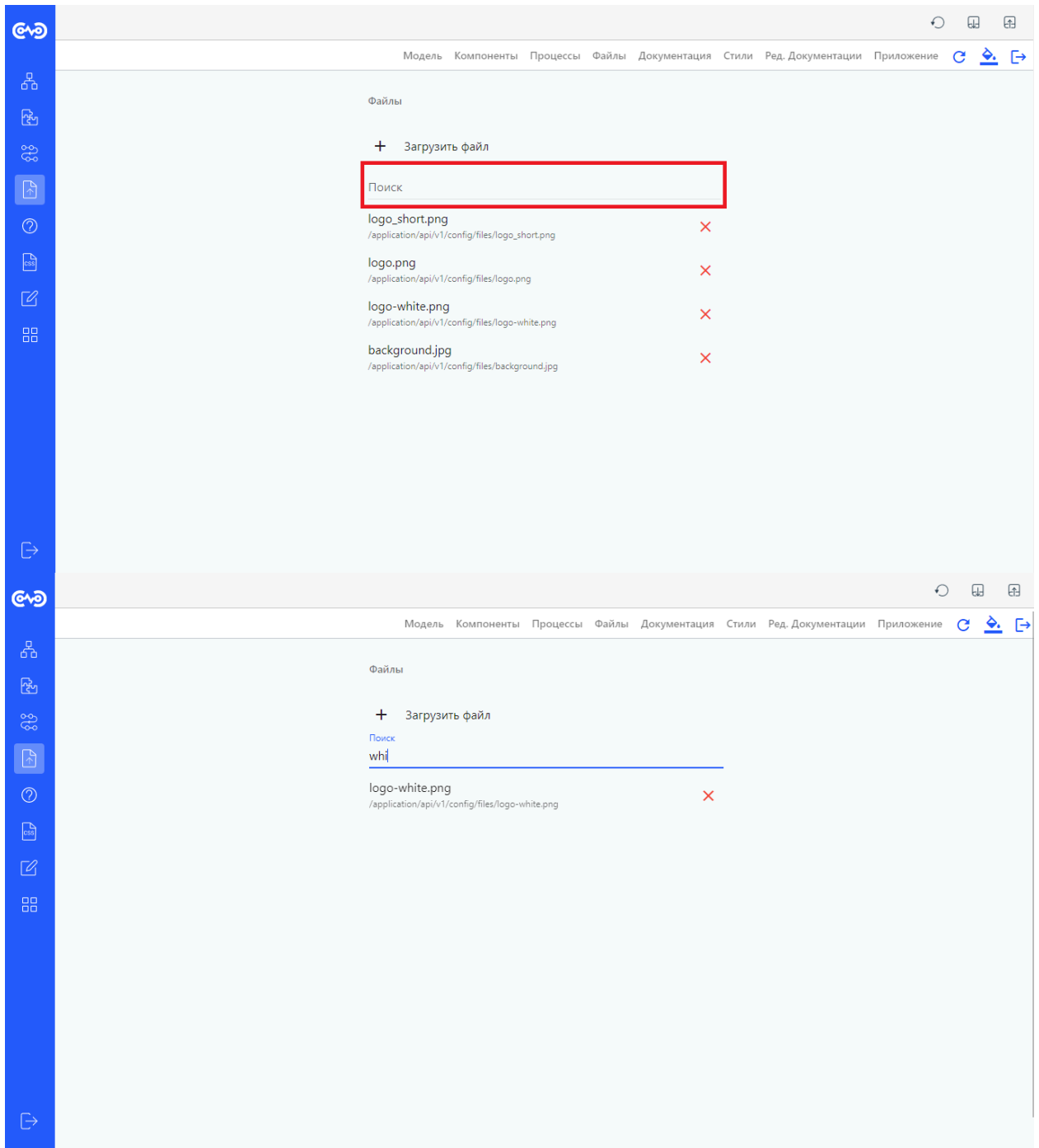
Файлы с русскими названиями загружаются, но происходит транслитерация.



Для использования этих файлов в компонентах потребуется url, который находится под каждым названием файла.

button2.png	/application/api/v1/config/files/button2.png	×
elementy vypadaiushchego spiska klient.jpg	/application/api/v1/config/files/elementy_vypadaiushchego_spiska_klient.jpg	×
button1.png	/application/api/v1/config/files/button1.png	×
dannye vypadaiushchego spiska klient.jpg	/application/api/v1/config/files/dannye_vypadaiushchego_spiska_klient.jpg	×

Если очень много файлов и необходимо найти тот, который нужен, то можно воспользоваться “Поиском”. Вы можете начать вводить название или часть названия, необходимые файлы найдутся и будут в списке, остальные не будут показываться.



Допустимый размер файла для загрузки в хранилище 50 мегабайт. Рекомендуемы форматы файлов для загрузки: jpg, png, jpeg, svg.

Стили интерфейса

Стили используются для визуального оформления бизнес-приложения.

Редактор имеет полную поддержку css, благодаря чему через селектор можно изменить стиль любого элемента.

Также есть набор переменных по умолчанию, к которым привязаны некоторые элементы интерфейса.

```
--padding-gutter: 22px;  
--padding: 10px;  
--padding-16: 1rem;  
--padding-12: 0.75rem;  
--padding-8: 0.5rem;  
--padding-4: 0.25rem;  
  
--color-primary: 38, 90, 252;  
--color-primary-contrast: 255, 255, 255;  
  
--color-accent: 255, 64, 129;  
--color-accent-contrast: 255, 255, 255;  
  
--color-warn: 255, 57, 45;  
--color-warn-contrast: 255, 255, 255;  
  
--text-base-dark-color: #000;  
--text-complement-dark-color: #000;  
--text-base-light-color: #000;  
--text-faint-light-color: #000;  
--text-faint-light-shift-down-color: #000;  
--text-faint-light-shift-up-color: #000;  
  
--primary-hover: #1F4EDE;  
--primary-press: #4786FF;  
--primary-press-background: #4786ff99;  
--primary-loading: #467CE3;  
--primary-stroke: #D9E1F9;  
--primary-light: #F3F6FB;  
--warn-hover: #D31004;  
--warn-press: #FF6E65;  
--warn-loading: #EC675E;  
--base-element-background-color: #FFF;  
--disable-color: #c4cdd5;  
--table-icon-color: #919EAB;  
--table-row-hover: #4786ff1f;  
--border-color: #919EAB52;  
--text-color-base: #212B36;  
--input-label-color: #637381;  
--scroll-color: #00004052;  
--scroll-background-color: #00000000;  
--breadcrumb-color: #00000061;
```


// Typography

```
--font: theme('fontFamily.sans');  
--font-weight-medium: 500;  
  
--font-caption-weight: 400;  
--font-caption-size: 12px;  
--font-caption-height: 20px;  
--font-caption-spacing: 0;  
  
--font-body-1-weight: 400;  
--font-body-1-size: 14px;  
--font-body-1-height: 20px;  
--font-body-1-spacing: -0.006em;  
  
--font-body-2-weight: 500;  
--font-body-2-size: 14px;  
--font-body-2-height: 24px;  
--font-body-2-spacing: -0.006em;  
  
--font-subheading-1-weight: 400;  
--font-subheading-1-size: 15px;  
--font-subheading-1-height: 24px;  
--font-subheading-1-spacing: -0.009em;  
  
--font-subheading-2-weight: 400;  
--font-subheading-2-size: 16px;  
--font-subheading-2-height: 28px;  
--font-subheading-2-spacing: -0.011em;  
  
--font-headline-weight: 400;  
--font-headline-size: 24px;  
--font-headline-height: 32px;  
--font-headline-spacing: -0.019em;  
  
--font-title-weight: 500;  
--font-title-size: 18px;  
--font-title-height: 26px;  
--font-title-spacing: -0.014em;  
  
--font-input-weight: 400;  
--font-input-size: 14px;  
--font-input-height: 1.125;  
--font-input-spacing: -0.006em;  
  
--font-checkbox-weight: 400;  
--font-checkbox-size: 14px;  
--font-checkbox-height: 21px;  
--font-checkbox-spacing: 0em;  
  
--font-radio-button-weight: 400;  
--font-radio-button-size: 14px;  
--font-radio-button-height: 21px;
```

```

--font-radio-button-spacing: -0.084px;

--font-button-weight: 500;
--font-button-size: 14px;
--font-button-height: 34px;
--font-button-spacing: normal;

--font-display-1-weight: 400;
--font-display-1-size: 34px;
--font-display-1-height: 40px;
--font-display-1-spacing: 0;

--font-display-2-weight: 400;
--font-display-2-size: 45px;
--font-display-2-height: 48px;
--font-display-2-spacing: -0.005em;

--font-display-3-weight: 400;
--font-display-3-size: 56px;
--font-display-3-height: 56px;
--font-display-3-spacing: -0.02em;

--font-display-4-weight: 300;
--font-display-4-size: 112px;
--font-display-4-height: 112px;
--font-display-4-spacing: -0.05em;

--font-caption: var(--font-caption-weight) var(--font-caption-size)/var(--font-caption-height) var(--font);
--font-body-1: var(--font-body-1-weight) var(--font-body-1-size)/var(--font-body-1-height) var(--font);
--font-body-2: var(--font-body-2-weight) var(--font-body-2-size)/var(--font-body-2-height) var(--font);
--font-subheading-1: var(--font-subheading-1-weight) var(--font-subheading-1-size)/var(--font-subheading-1-height) var(--font);
--font-subheading-2: var(--font-subheading-2-weight) var(--font-subheading-2-size)/var(--font-subheading-2-height) var(--font);
--font-headline: var(--font-headline-weight) var(--font-headline-size)/var(--font-headline-height) var(--font);
--font-title: var(--font-title-weight) var(--font-title-size)/var(--font-title-height) var(--font);
--font-display-1: var(--font-display-1-weight) var(--font-display-1-size)/var(--font-display-1-height) var(--font);
--font-display-2: var(--font-display-2-weight) var(--font-display-2-size)/var(--font-display-2-height) var(--font);
--font-display-3: var(--font-display-3-weight) var(--font-display-3-size)/var(--font-display-3-height) var(--font);
--font-display-4: var(--font-display-4-weight) var(--font-display-4-size)/var(--font-display-4-height) var(--font);
--font-button: var(--font-button-weight) var(--font-button-size)/var(--font-button-height) var(--font);
--font-input: var(--font-input-weight) var(--font-input-size)/var(--font-input-height) var(--font);
--font-checkbox: var(--font-checkbox-weight) var(--font-checkbox-size)/var(--font-checkbox-height) var(--font);

```

```
--font-radio-button: var(--font-radio-button-weight) var(--font-radio-button-size)/var(--font-radio-button-height) var(--font);
```

// Transitions

```
--trans-ease-in-out: all var(--trans-ease-in-out-duration) var(--trans-ease-in-out-timing-function);  
--trans-ease-in-out-duration: #{ $swift-ease-in-out-duration };  
--trans-ease-in-out-timing-function: #{ $swift-ease-in-out-timing-function };  
--trans-ease-out: all var(--trans-ease-out-duration) var(--trans-ease-out-timing-function);  
--trans-ease-out-duration: #{ $swift-ease-out-duration };  
--trans-ease-out-timing-function: #{ $swift-ease-out-timing-function };  
--trans-ease-in: all var(--trans-ease-in-duration) var(--trans-ease-in-timing-function);  
--trans-ease-in-duration: #{ $swift-ease-in-duration };  
--trans-ease-in-timing-function: #{ $swift-ease-in-timing-function };  
--trans-shadow-duration: #{ $mat-elevation-transition-duration };  
--trans-shadow-timing-function: #{ $mat-elevation-transition-timing-function };  
--trans-shadow: box-shadow var(--trans-shadow-duration) var(--trans-shadow-timing-function);
```

```
--text-color: #{ $dark-primary-text };  
--text-color-light: #{ $slight-primary-text };  
--text-secondary: #{ $dark-secondary-text };  
--text-secondary-light: #{ $slight-secondary-text };  
--text-hint: #{ $dark-disabled-text };  
--text-hint-light: #{ $slight-disabled-text };
```

// Foreground

```
--foreground-base: 0, 0, 0;  
--foreground-base-alpha: 1;  
--foreground-divider: 82, 63, 105;  
--foreground-divider-alpha: 0.06;  
--foreground-dividers: 0, 0, 0;  
--foreground-dividers-alpha: 1;  
--foreground-disabled: 0, 0, 0;  
--foreground-disabled-alpha: 1;  
--foreground-disabled-button: 0, 0, 0;  
--foreground-disabled-button-alpha: .26;  
--foreground-text-disabled: 0, 0, 0;  
--foreground-text-disabled-alpha: 1;  
--foreground-elevation: 82, 63, 104;  
--foreground-elevation-alpha: 0.04;  
--foreground-hint-text: 0, 0, 0;  
--foreground-hint-text-alpha: 0.26;  
--foreground-secondary-text: 0, 0, 0;  
--foreground-secondary-text-alpha: .54;  
--foreground-icon: 0, 0, 0;  
--foreground-icon-alpha: 0.54;  
--foreground-icons: 0, 0, 0;  
--foreground-icons-alpha: 0.54;  
--foreground-text: 0, 0, 0;  
--foreground-text-alpha: 0.87;  
--foreground-slider-min: 0, 0, 0;  
--foreground-slider-min-alpha: 0.87;  
--foreground-slider-off: 0, 0, 0;  
--foreground-slider-off-alpha: 0.26;
```

```
--foreground-slider-off-active: 0, 0, 0;  
--foreground-slider-off-active-alpha: 0.38;  
--foreground-disabled-text: 0, 0, 0;  
--foreground-disabled-text-alpha: 0.38;
```

// Background

```
--background-status-bar: 224, 224, 224;  
--background-status-bar-alpha: 1;  
--background-app-bar: 235, 235, 238;  
--background-app-bar-alpha: 1;  
--background-background: 250, 250, 250;  
--background-background-alpha: 1;  
--background-hover: 0, 0, 0;  
--background-hover-alpha: 0.04;  
--background-card: 255, 255, 255;  
--background-card-alpha: 1;  
--background-dialog: 255, 255, 255;  
--background-dialog-alpha: 1;  
--background-disabled-button: 0, 0, 0;  
--background-disabled-button-alpha: 0.12;  
--background-raised-button: 255, 255, 255;  
--background-raised-button-alpha: 1;  
--background-focused-button: 0, 0, 0;  
--background-focused-button-alpha: 1;  
--background-selected-button: 224, 224, 224;  
--background-selected-button-alpha: 1;  
--background-disabled-button-toggle: 189, 189, 189;  
--background-disabled-button-toggle-alpha: 1;  
--background-unselected-chip: 224, 224, 224;  
--background-unselected-chip-alpha: 1;  
--background-disabled-list-option: 238, 238, 238;  
--background-disabled-list-option-alpha: 1;  
--background-tooltip: 97, 97, 97;  
--background-tooltip-alpha: 1;  
--background-base: 242, 245, 247;  
--background-base-alpha: 1;
```

// Elevation

```
--elevation-default: var(--elevation-z6);  
--elevation-z0: none;  
--elevation-z1: #{g-elevation(1)};  
--elevation-z2: #{g-elevation(2)};  
--elevation-z3: #{g-elevation(3)};  
--elevation-z4: #{g-elevation(4)};  
--elevation-z5: #{g-elevation(5)};  
--elevation-z6: #{g-elevation(6)};  
--elevation-z7: #{g-elevation(7)};  
--elevation-z8: #{g-elevation(8)};  
--elevation-z9: #{g-elevation(9)};  
--elevation-z10: #{g-elevation(10)};  
--elevation-z11: #{g-elevation(11)};  
--elevation-z12: #{g-elevation(12)};  
--elevation-z13: #{g-elevation(13)};
```

```
--elevation-z14: #{g-elevation(14)};
--elevation-z15: #{g-elevation(15)};
--elevation-z16: #{g-elevation(16)};
--elevation-z17: #{g-elevation(17)};
--elevation-z18: #{g-elevation(18)};
--elevation-z19: #{g-elevation(19)};
--elevation-z20: #{g-elevation(20)};
```

// Sidenav

```
--sidenav-width: 264px;
--sidenav-collapsed-width: 72px;
--sidenav-background: var(--primary-hover);
--sidenav-color: white;
--sidenav-toolbar-logo-height: auto !important;
--sidenav-toolbar-logo-width: 5rem;
--sidenav-item-padding: 1em;
--sidenav-toolbar-background: var(--primary-hover);
--sidenav-item-background-active: var(--primary-press-background);
--sidenav-item-color: var(--base-element-background-color);
--sidenav-item-color-active: var(--base-element-background-color);
--sidenav-item-icon-color: var(--base-element-background-color);
--sidenav-item-icon-color-active: var(--base-element-background-color);
--sidenav-item-icon-gap: 0.714em;
--sidenav-item-icon-size: 24px;
--sidenav-item-border-color: transparent;
--sidenav-item-border-color-active: rgb(var(--color-primary));
--sidenav-item-dropdown-background: var(--primary-hover);
--sidenav-item-dropdown-background-hover: var(--primary-hover);
--sidenav-item-dropdown-gap: 12px;
--sidenav-item-margin-top: 0.14em;
--sidenav-item-level-1-active-before-font-size: 1.714em;
--sidenav-item-level-1-before-content: "•";
--sidenav-item-level-1-before-font-size: 1em;
--sidenav-item-level-1-before-position: absolute;
--sidenav-item-level-1-before-margin-left: -1.7rem;
--sidenav-subheading-color: var(--base-element-background-color);
--sidenav-subheading-font-weight: 700;
--sidenav-subheading-height: 3.714em;
--sidenav-item-width: 240px;
--sidenav-item-margin-left: 0.57em;
--sidenav-item-min-height: 2.857em;
--sidenav-item-border-radius: var(--border-radius);
--sidenav-item-transition: var(--trans-ease-out), font 0ms linear;
--sidenav-item-active-font-weight: bold;
```

// Toolbar

```
--toolbar-height: 4em;
--toolbar-background: white;
--toolbar-color: #{dark-primary-text};
--toolbar-icon-color: rgb(var(--color-primary));
--toolbar-button-color: rgba(var(--color-primary), 1);
--toolbar-button-border-color: rgba(var(--foreground-divider), var(--foreground-divider-alpha));
--toolbar-button-font: var(--font-button);
```

```

--toolbar-button-letter-spacing: var(--font-button-spacing);
--toolbar-button-text-transform: none;
--toolbar-button-padding: 0 0.6rem;
--toolbar-notifications-dropdown-background: rgb(var(--color-primary));
--toolbar-notifications-dropdown-color: rgb(var(--color-primary-contrast));
--toolbar-user-dropdown-background: var(--base-element-background-color);
--toolbar-user-dropdown-color: var(--text-color-base);
--toolbar-user-title-button-color: var(--toolbar-color);
--toolbar-user-title-button-border-color: var(--toolbar-button-border-color);
--toolbar-user-title-button-font: var(--toolbar-button-font);
--toolbar-user-title-button-letter-spacing: var(--toolbar-button-letter-spacing);
--toolbar-user-title-button-text-transform: var(--toolbar-button-text-transform);
--toolbar-user-title-button-padding: var(--toolbar-button-padding);
--toolbar-user-dropdown-width: 220px;
--toolbar-user-dropdown-description-display: none;
--toolbar-user-dropdown-chevron-display: none;
--toolbar-user-dropdown-avatar-display: none;
--toolbar-user-dropdown-label-color: var(--text-color-base);
--toolbar-user-dropdown-label-hover-color: var(--text-color-base);
--toolbar-user-dropdown-icon-color: var(--text-color-base);
--toolbar-user-dropdown-footer-border-top: none;
--toolbar-user-dropdown-footer-background: var(--base-element-background-color);
--toolbar-user-dropdown-notification-border-top: none;
--toolbar-user-dropdown-notification-padding: 0.571em 0 0.571em 1.429em;
--toolbar-user-dropdown-footer-button-color: var(--text-color-base);
--toolbar-user-dropdown-footer-button-border: 1px solid var(--primary-stroke);
--toolbar-user-dropdown-footer-button-width: 100%;
--toolbar-user-dropdown-footer-button-padding: 0;
--toolbar-user-dropdown-footer-button-hover-color: rgb(var(--color-primary));
--toolbar-user-dropdown-footer-button-hover-border: 1px solid rgb(var(--color-primary));
--toolbar-user-dropdown-footer-button-hover-background-color: var(--base-element-background-color);
--toolbar-user-dropdown-header-border-bottom: 1px solid var(--border-color);
--toolbar-user-dropdown-header-padding: 1.143em 1.429em 1.214em 1.429em;
--toolbar-user-dropdown-header-margin-bottom: 0.571em;
--toolbar-user-dropdown-icon-margin-right: 1.143em;

```

// Navigation

```

--navigation-height: 64px;
--navigation-background: rgba(var(--background-card), var(--background-card-alpha));
--navigation-color: var(--text-secondary);

```

// Footer

```

--footer-height: 56px;
--footer-z-index: 100;
--footer-background: rgba(var(--background-card), var(--background-card-alpha));
--footer-color: var(--text-color);
--footer-elevation: 0 -10px 30px 0 rgba(82, 63, 104, .06);

```

// Misc

```

--default-icon-size: 24px;
--border-radius: 8px;

```

// Datepicker

```
--datepicker-input-suffix-top: 0.357em;  
--datepicker-toggle-color: var(--text-color-base);  
--datepicker-input-label-margin-top: -0.4em;  
--datepicker-input-flex-align-items: initial;
```

// Scroll

```
--webkit-scrollbar-width: 12px;  
--webkit-scrollbar-width: 8px;  
--webkit-scrollbar-background-color: var(--scroll-background-color);  
--webkit-scrollbar-hover-background-color: var(--scroll-background-color);  
--webkit-scrollbar-thumb-border: 2px solid transparent;  
--webkit-scrollbar-thumb-box-shadow: inset 0 0 0 12px var(--scroll-color);  
--webkit-scrollbar-thumb-border-radius: var(--webkit-scrollbar-width);  
--webkit-scrollbar-thumb-active-box-shadow: inset 0 0 0 12px var(--scroll-color);  
--webkit-scrollbar-thumb-active-border-radius: var(--webkit-scrollbar-width);
```

// Card

```
--card-border-radius: var(--border-radius);  
--card-box-shadow: none;
```

// Button

```
--button-primary-background: rgba(var(--color-primary), 1);  
--button-warn-background: rgba(var(--color-warn), 1);  
--button-accent-background: rgba(var(--color-accent), 1);  
  
--background-disabled-button: var(--disable-color);  
--button-disabled-background: var(--background-disabled-button);  
--button-disabled-color: rgba(var(--foreground-disabled-button), var(--foreground-disabled-button-alpha));  
  
--button-border-radius: var(--border-radius);  
--button-padding: var(--padding) var(--padding-gutter);  
--button-icon-margin-right: 4px;  
--button-primary-hover-background: var(--primary-hover);  
--button-line-height: 36px;
```

// Button set

```
--button-set-horizontal-direction-spacing: var(--padding-8);  
--button-set-vertical-direction-spacing: var(--padding-8);
```

// Input

```
--input-standard-border: none;  
--input-standard-box-shadow: none;  
--input-fill-border-radius: var(--border-radius);  
--input-fill-background: var(--base-element-background-color);  
--input-fill-accent-background: #ffe4c4;  
--input-fill-line-height: 1.5em;  
--input-fill-color: var(--text-color-base);  
--input-fill-padding: 0 1em 0 1em;  
--input-fill-error-color: rgba(var(--color-warn), 1);
```

```

--input-fill-error-font-size: 10px;
--input-fill-error-before-display: none;
--input-fill-subscript-wrapper-padding: 0;
--input-fill-underline: none;
--input-fill-label-color: var(--input-label-color);
--input-fill-focused-label-color: rgba(var(--color-primary), 1);
--input-fill-invalid-label-color: rgba(var(--color-warn), 1);
--input-fill-infix-padding: 0.714em 0 0.785em 0;
--input-fill-disabled-background: var(--base-element-background-color);
--input-fill-disabled-text: rgba(var(--foreground-disabled-text), var(--foreground-disabled-text-alpha));
--input-fill-disabled-label: rgba(var(--foreground-disabled-text), var(--foreground-disabled-text-alpha));
--input-fill-label-wrapper-top: 0;
--input-fill-label-wrapper-left: 0px;
--input-fill-label-transform: transform 400ms cubic-bezier(0.25, 0.8, 0.25, 1),color 400ms cubic-bezier(0.25, 0.8, 0.25, 1),width 400ms cubic-bezier(0.25, 0.8, 0.25, 1),left 400ms cubic-bezier(0.25, 0.8, 0.25, 1);
--input-fill-label-width: auto;
--input-fill-focus-label-transform: translateY(-1.4em) scale(0.75);
--input-fill-focus-label-width: auto;
--input-fill-suffix-color: var(--text-color-base);
--input-fill-suffix-margin: 0;
--input-fill-prefix-color: inherit;
--input-fill-prefix-margin: 0;
--input-fill-prefix-align-self: center;
--input-fill-select-arrow-wrapper-transform: none;
--input-fill-label-background: var(--base-element-background-color);
--input-fill-label-padding: 0 0.25em;
--input-fill-label-margin-top: -0.3em !important;
--input-fill-label-left: -0.25em;
--input-fill-infix-border-top: none;
--input-fill-label-wrapper-padding: 0;
--input-fill-label-wrapper-overflow: visible !important;
--input-fill-focus-label-display: block;
--input-fill-invalid-border: 1px solid rgb(var(--color-warn));
--input-fill-border: 1px solid var(--primary-stroke);
--input-fill-focused-border: 1px solid var(--primary-hover);

--input-standard-border: none;
--input-standard-box-shadow: none;
--input-standard-border-radius: 0;
--input-standard-background: none;
--input-standard-line-height: 1.5em;
--input-standard-color: #000;
--input-standard-margin: 0;
--input-standard-padding: 0 0 0 0;
--input-standard-error-color: rgba(var(--color-warn), 1);
--input-standard-error-font-size: 10px;
--input-standard-error-before-display: none;
--input-standard-subscript-wrapper-padding: 0;
--input-standard-invalid-border: none;
--input-standard-focused-border: none;

```



```

--input-standard-focused-box-shadow: none;
--input-standard-underline: block;
--input-standard-focused-label-color: rgba(var(--color-primary), 1);
--input-standard-invalid-label-color: rgba(var(--color-warn), 1);
--input-standard-infix-padding: 0.5em 0;
--input-standard-infix-border-top: 0.84375em solid transparent;
--input-standard-disabled-background: none;
--input-standard-disabled-text: rgba(var(--foreground-disabled-text), var(--foreground-disabled-text-alpha)));
--input-standard-disabled-label: rgba(var(--foreground-disabled-text), var(--foreground-disabled-text-alpha)));
--input-standard-label-wrapper-top: -0.84375em;
--input-standard-label-wrapper-left: 0px;
--input-standard-label-color: rgba(var(--foreground-secondary-text), 0.6);
--input-standard-label-transform: transform 400ms cubic-bezier(0.25, 0.8, 0.25, 1),color 400ms cubic-bezier(0.25, 0.8, 0.25, 1),width 400ms cubic-bezier(0.25, 0.8, 0.25, 1);
--input-standard-label-width: 100%;
--input-standard-focus-label-transform: translateY(-1.34375em) scale(0.75);
--input-standard-focus-label-width: 133.3333333333333%;
--input-standard-suffix-color: inherit;
--input-standard-suffix-margin: 0;
--input-standard-prefix-color: inherit;
--input-standard-prefix-margin: 0;

```

// Tabs

```

--tabs-header-border-bottom: 1px solid rgba(var(--foreground-divider), var(--foreground-divider-alpha));
--tabs-header-buttons-space: 0;
--tabs-header-button-color: rgba(var(--foreground-text), var(--foreground-text-alpha)));
--tabs-header-button-opacity: .6;
--tabs-header-button-active-color: inherit;
--tabs-header-button-active-opacity: 1;
--tabs-header-button-active-background: none;
--tabs-header-button-active-border-radius: none;
--tabs-header-inkbar-background: rgba(var(--color-primary), 1);
--tabs-header-inkbar-height: 2px;

```

// Checkbox

```

--checkbox-container-height: 16px;
--checkbox-container-width: 16px;
--checkbox-label-font: var(--font-checkbox);
--checkbox-color: var(--text-color-base);
--checkbox-frame-border: 2px solid rgba(var(--foreground-secondary-text), var(--foreground-secondary-text-alpha));
--checkbox-frame-box-shadow: none;

```

// Checkbox group

```

--checkbox-group-horizontal-direction-spacing: 1em;
--checkbox-group-vertical-direction-spacing: 0.5em;

```

// Radio

```

--radio-button-width: 20px;
--radio-button-height: 20px;

```

```

--radio-button-container-background: none;
--radio-button-outer-circle-border: 2px solid rgba(var(--foreground-secondary-text), var(--foreground-secondary-text-alpha));
--radio-button-outer-circle-box-shadow: none;
--radio-button-checked-inner-circle-transform: scale(0.5);
--radio-button-checked-outer-circle-border-width: 2px;
--radio-button-inner-circle-background-color: inherit;
--radio-button-label-color: inherit;
--radio-button-label-padding: 0 0 0 8px;
--radio-button-label-font: inherit;
--radio-horizontal-direction-spacing: 1em;
--radio-vertical-direction-spacing: 0.5em;

```

// Page layout

```

--page-layout-panel-background: initial;
--page-layout-header-margin: auto;
--page-layout-content-max-width: 100%;
--page-layout-header-padding: var(--padding) var(--padding-gutter) 0 var(--padding-gutter);
--page-layout-content-padding: var(--padding) var(--padding-gutter);
--page-layout-header-white-breadcrumb-color-color: var(--breadcrumb-color);
--page-layout-header-white-breadcrumb-color-hover-color: var(--breadcrumb-color);
--page-layout-header-white-title-color: var(--text-color-base) !important;
--page-layout-header-white-title-font: 700 1.428em/2.142em var(--font);
--page-layout-header-white-title-font-style: normal;
--page-layout-header-white-title-text-align: left;

```

// Select

```

--select-dropdown-margin-top: 36px;
--select-dropdown-box-shadow: 0 0 0 1px var(--primary-stroke);
--select-dropdown-border-radius: var(--border-radius);
--select-dropdown-item-height: 3.142em;
--select-dropdown-item-selected-background: var(--primary-stroke);
--select-dropdown-item-selected-color: rgb(var(--color-primary));
--select-dropdown-item-hover-background: var(--primary-light);
--select-dropdown-item-border-radius: var(--border-radius);
--select-dropdown-item-color: var(--text-color-base);
--select-dropdown-item-margin: 0.285em 0.571em;
--select-dropdown-item-padding: 0 0.571em;

```

// Table

```

--table-search-operations-buttons-border: 1px solid var(--primary-stroke);
--table-search-operations-buttons-line-height: 2.45em;
--table-search-operations-buttons-padding: 0 1.071em;
--table-search-operations-buttons-color: var(--text-color-base);
--table-search-operations-buttons-icon-color: rgb(var(--color-primary));
--table-datatable-mat-menu-buttons-color: var(--text-color-base);
--table-datatable-mat-menu-buttons-icon-color: rgb(var(--color-primary));
--table-datatable-border-top: none;
--table-datatable-footer-border-top: none;
--table-datatable-body-row-cell-padding: 0.714em 1.357em;
--table-datatable-row-cell-border-bottom: none;
--table-datatable-body-row-cell-menu-trigger-color: var(--table-icon-color);
--table-input-input-fill-infix-padding: 5px 0 8px 0;

```

```

--table-input-input-fill-padding: 0 0.857em 0 0.857em;
--table-search-input-input-fill-padding: 0 3em 0 0.429em;
--table-search-input-prefix-width: auto;
--table-input-suffix-display: block;
--table-search-input-infix-border-right: 1px solid var(--primary-stroke);
--table-search-input-infix-focus-border-right: 1px solid rgb(var(--color-primary));
--table-header-input-width: 236px;
--table-input-label-margin: -0.3em;
--table-input-label-padding: 0;
--table-input-label-top: 10px;
--table-input-label-left: 0.286em;
--table-header-input-prefix-button-vertical-align: middle;
--table-header-panel-input-prefix-icon-font-size: 150%;
--table-header-panel-input-prefix-icon-margin-left: -0.571em;
--table-header-input-prefix-icon-button-disable-padding: 0 0;
--table-header-input-prefix-icon-button-disable-display: inline;
--table-header-input-prefix-icon-button-disable-color: var(--table-icon-color);
--table-header-panel-padding: 0.2em 0.5em;
--table-header-filters-position: absolute;
--table-header-filters-left: 201px;
--table-header-filters-color: rgb(var(--color-primary));
--table-header-filters-height: 2.45em;
--table-header-filters-top: 0.5em;
--table-header-filters-width: 3em;
--table-header-filters-border-radius: 0;
--table-search-input-width: 236px;
--table-input-focus-label-display: var(--input-fill-focus-label-display);
--table-input-focus-label-transform: var(--input-fill-focus-label-transform);
--table-input-label-transform: var(--input-fill-label-transform);
--table-datatable-body-row-hover-background-color: var(--table-row-hover);
--table-datatable-header-border-bottom: none;
--table-datatable-header-height: auto;
--table-datatable-header-background-color: var(--primary-light);
--table-datatable-header-cell-padding: 0.714em 1.2rem;
--table-datatable-header-icon-sort-unset-display: none;
--table-datatable-header-cell-background-color: var(--primary-light);
--table-datatable-header-row-right-background-color: var(--primary-light);
--table-datatable-footer-pager-item-hover-border-radius: 50%;
--table-datatable-footer-pager-item-active-border-radius: 50%;
--table-datatable-footer-pager-item-active-background-color: var(--primary-stroke);
--table-datatable-footer-pager-item-active-color: rgb(var(--color-primary));
--table-datatable-footer-pager-item-active-font-weight: 400;
--table-datatable-footer-pager-item-min-width: 24px;
--table-datatable-footer-pager-item-height: 24px;
--table-datatable-footer-pager-item-font-size: 1em;
--table-datatable-footer-pager-item-line-height: 1.714em;
--table-datatable-footer-pager-item-color: var(--text-color-base);
--table-datatable-footer-pager-item-display: flex;
--table-datatable-footer-pager-item-align-items: center;
--table-datatable-footer-pager-item-justify-content: center;
--table-datatable-footer-pager-item-padding: 0;
--table-datatable-footer-pager-item-margin: 3px;
--table-header-filters-line-height: 0;

```

```
--table-search-input-prefix-icon-height: 2.45em;
--table-search-input-suffix-display: none;
--table-search-input-focus-label-display: none;
--table-filters-input-focus-label-display: block;
--table-filters-input-focus-label-transform: translateY(-1em) scale(0.75);
--table-filters-input-label-transform: var(--input-fill-label-transform);
--table-filters-input-label-wrapper-top: -5px;
--table-header-filters-panel-padding: 0.2rem 0.5rem 0.5rem 0.5rem;
--table-datatable-row-active-background-color: var(--table-row-hover);
--table-datatable-row-active-first-cell-box-shadow: inset 2px 0 0 0 rgb(var(--color-primary));
--table-datatable-footer-pager-item-width: 1.714em;
```

// Info

```
--info-message-border-radius: var(--border-radius);
--info-message-padding: var(--padding-16);
--info-message-gap: var(--padding-8);
--info-message-content-font-size: 16px;
```

// Stepper

```
--stepper-default-color: rgba(0, 0, 0, 0.3);
--stepper-default-background: rgba(0, 0, 0, 0.1);
--stepper-checked-color: rgb(var(--color-primary));
--stepper-icon-color: #fff;
--stepper-distance-length: 100%;
--stepper-distance-width: var(--stepper-item-border-width);
--stepper-item-size: 30px;
--stepper-item-border-width: 2px;
--stepper-item-font-size: 14px;
--stepper-item-weight: 500;
--stepper-border-radius: 50%;
```