BIVPlaform

Документация, содержащая описание функциональных характеристик программного обеспечения

01.10.2026

Функциональные характеристики программы

BIVPlatform (Платформа BIV) — это набор инфраструктурных модулей и компонентов, готовых шаблонов для разработчиков, а также свод регламентов и правил для создания и развития информационных систем для финансовых организаций в микросервисной архитектуре.

Платформа BIV создана и используется для:

- повышения скорости и эффективности разработки типовых задач;
- унификации используемых подходов, компонентов, технологий (проще передавать решения между командами, привлекать новых разработчиков);
- применения уже готовых инфраструктурных и общесистемных модулей.

Платформа BIV разработана на языке Java с использованием фреймворков Quarkus, Angular.

Фреймворк Quarkus используется для построения бэкенда. Quarkus собран из лучших библиотек и стандартов Java и предназначен для создания современных императивных и реактивных Java-приложений в первую очередь для облачной платформы Kubernetes.

Фреймворк Angular используется для разработки фронтенда. Angular хорошо подходит для создания крупных и надежных корпоративных приложений благодаря строгой архитектуре, языку TypeScript, а также своему набору инструментов, API и библиотек для упрощения и оптимизации процесса разработки.

В рамках Платформы BIV разработаны следующие библиотеки и расширения:

- Базовые библиотеки общего назначения.
  - Библиотека rest-common (бэкенд) обеспечивает поддержку спецификации запроса и ответа для rest-api микросервисов, обработку и проброс исключительных ситуаций.
  - Библиотека crypto-service (бэкенд) предоставляет методы по шифрованию данных при взаимодействии фронтенда и бэкенда.
  - Библиотека permission-base (бэкенд) декларирует базовые интерфейсы по авторизации. Данные интерфейсы могут быть реализованы либо библиотеками авторизации, входящими в состав платформы, либо есть возможность реализовать их иначе.
- Расширение entity-info (бэкенд).
  - Расширяет описание модели данных (сущностей) в ORM информацией на естественном языке в терминах предметной области.
  - Расширенное описание предметной области позволяет бизнес-пользователю самостоятельно конфигурировать систему.
  - На базе данного расширения строятся все остальные библиотеки, расширения и сервисы.
- Расширение entity-data (бэкенд).
  - Обеспечивает работу с данными на основе расширенного описания модели данных с учётом прав пользователя.
  - Формирует и исполняет запросы к базе данных.
  - Реализует CRUD-операции над сущностями с возможностью их дополнения в точках расширения.
- Каркас веб-приложения (фронтенд).
  - Каркас является основой, которая предоставляет базовую функциональность (стартовая страница, главное меню, профиль пользователя и т. п.), а также объединяет микрофронтенды различных сервисов.
- Компоненты для построения пользовательского интерфейса (фронтенд).
  - UI-элементы с встроенной логикой взаимодействия с моделью данных.
- Библиотека entity-journal (бэкенд и фронтенд) — конструктор табличного

представление данных.
    - Позволяет пользователю настраивать представление табличных данных на основании расширенной распределённой модели данных.
    - В рамках представления определяются состав колонок, формат отображаемых данных, их сортировка. Дополнительно определяются критерии выборки данных, которые должен указать в ходе работы пользователь.
    - Пользователь может создать несколько представлений и переключаться между ними.
    - Отображение данных осуществляется с учётом прав пользователя.
- Библиотека Server Driven User Interface — SDI (бэкенд и фронтенд).
    - Формирование содержимого страницы пользовательского интерфейса на стороне бэкенда с помощью простой json-разметки и отображение этого содержимого на фронтенде. Это разметка более высокого уровня, чем HTML. Разметка не содержит специфических деталей реализации для конкретной платформы на стороне фронтенда. А фронтендом могут выступать не только веб, но и мобильные приложения или толстые десктопные клиенты. Фронетенд знает как отрисовывать каждую компоненту нативным способом. Стилистика элементов пользовательского интерфейса определяется на стороне фронтенда при создании каркаса приложения.
    - Страницы пользовательского интерфейса могут храниться как в самом прикладном сервисе (отсутствует возможность их модификации без пересборки прикладного сервиса), так и в базе данных (можно оперативно менять страницы «на лету»).
    - Библиотека может автоматически генерировать пользовательский интерфейс (низкой и средней степени сложности) на основе расширенного описания модели данных. При этом есть возможность внесения уточнений в процесс генерации через json-разметку.
    - Конструктор экранных форм с возможностью автоматической генерации пользовательского интерфейса по модели данных — библиотека Server driven user interface.
- Библиотека авторизации (бэкенд).
    - Обеспечивает вычисление эффективных прав пользователя на основе правил и контекста исполнения.
    - Реализует базовые интерфейсы авторизации и через них предоставляет результаты вычисления эффективных прав для дальнейшего контроля операций.
- Библиотека state-machine — машина состояний, основанная на расширенной модели данных.
    - «Навешивается» на сущность из расширенной модели данных.
    - Позволяет описать граф переходов состояний и триггеры перехода.
    - Осуществляет перевод объекта сущности между состояниями согласно графу, выполнение триггеров состояния.
- Библиотека refbook для описания и хранения произвольных справочных данных.
    - Позволяет эффективно настраивать НСИ без перекомпиляции исходного кода системы. Дополняет расширенную модель данных.
    - Определят микросервис, являющийся владельцем справочника, и микросервисы-потребители данных справочника.
    - Встроенный механизм репликации данных позволяет синхронизировать содержимое НСИ между микросервисами.
    - Предоставляет API для использования данных НСИ в информационной системе.
- Библиотека, реализующая работу с файлами
    - Позволяет загружать, скачивать, отображать файлы, просматривать превью, удалять файлы

    - Предоставляет API для работы с файлами
    - В качестве конечного хранилища используется DFS система хранения файлов
- Библиотека, реализующая работу с параметрами приложения
    - С помощью параметров приложения можно изменять поведение системы под конкретные нужды. Реакция на параметр настраивается и связывается с определённым параметром программистом, после чего значение можно получить через пользовательский интерфейс или программно.
    - Тип значения параметров приложения может быть текстовым, числовым, логическим, временным, объектом (JSON) или массивом (JSON-список). Примеры параметров: текущий период работы, текущая дата работы, текущий план счетов, максимальное число полей и т.д.
    - В дополнение предусмотрены три среза (уровня) параметров. По умолчанию все три среза GLOBAL — глобальные, т.е. параметр по умолчанию является глобальным. Ограничений на наименование среза нет. Срезы с одинаковым наименованием рекомендуется задавать на одном уровне, например, первый срез для всех параметров = USER или GLOBAL.
- Сервисы аудита
    - Аудит представляет собой систему фиксации и хранения событий, происходящих в информационной системе. Компонента аудита обеспечивает регистрацию действий пользователей, изменений данных и системных операций для последующего анализа, расследования инцидентов и соблюдения требований регуляторов.
- Сервисы уведомлений
    - Сервис уведомлений предназначен для универсальной обработки, хранения и доставки уведомлений в интерфейс пользователя.
    - Сервис абстрагирован от бизнес-логики систем-отправителей и фокусируется на приёме запросов на отправку, маршрутизации, доставке уведомлений и предоставлении интерфейсов для их отображения и обработки.
    - Взаимодействие с сервисом осуществляется через библиотеку.

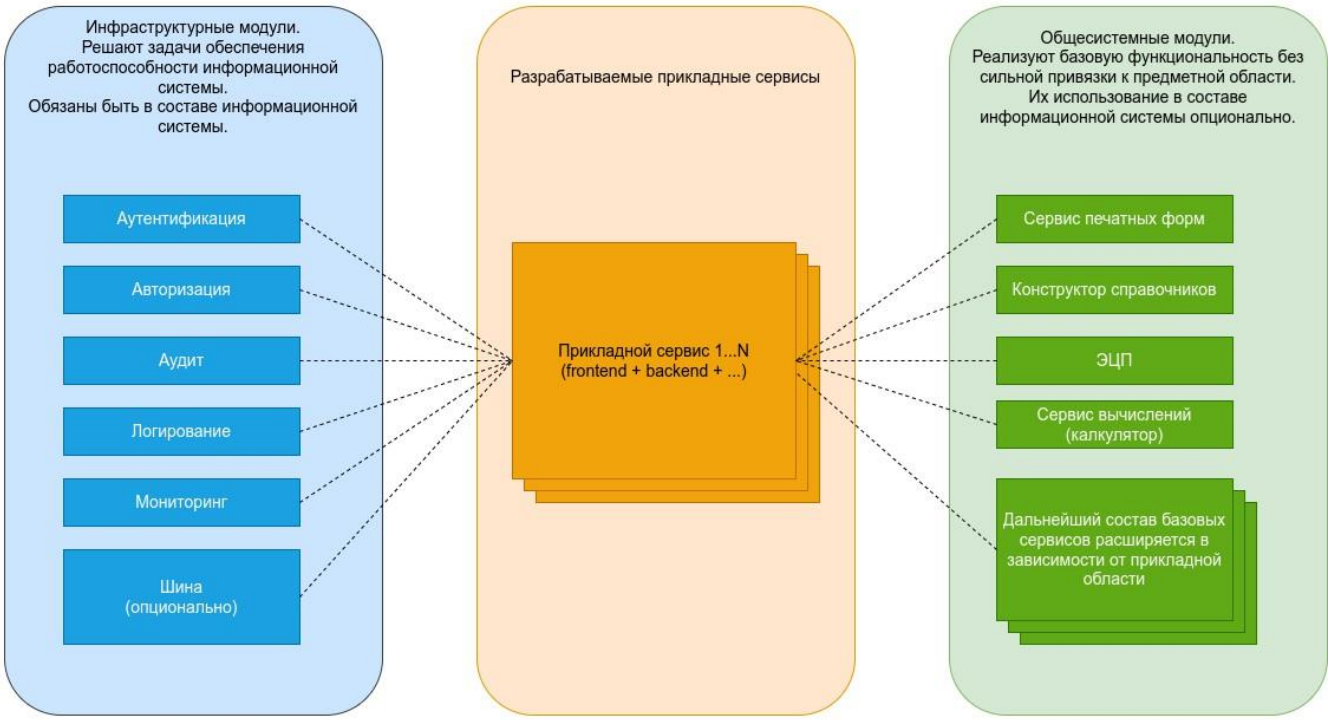
Платформа BIV включает в себя следующие общесистемные сервисы:

- Сервис авторизации
    - Модуль Авторизации построен на основе подхода “Policy as Code” / “Declarative Policy”, в котором для определения прав используется специальный декларативный язык Cedar. Такой подход отличается высокой гибкостью, взаимодействует с моделью данных прикладного сервиса, учитывает контекст исполнения правил.
- Модуль Формирования печатных форм.
    - Предназначен для автоматизации всей цепочки операций по формированию печатных документов, а именно:
    - Имеет пользовательский интерфейс по созданию и управлению шаблонами.
    - При формировании печатных документов обеспечивает подстановку данных и трансформацию документа на основании инструкций в шаблоне.
    - Обеспечивает контроль и разграничение доступа к шаблонам и выходным документам.
    - Предоставляет API для встраивания в информационные системы.
    - Шаблоны печатных форм создаются в офисном пакете (MS Office / Open Office) с помощью специальных управляющих конструкций (разметки) и могут быть в форматах MS Office – (DOCX, XLSX) Open Office (ODT, ODS).
    - Печатная форма может иметь изменяемую структуру (переменное количество разделов, скрываемые / динамические параграфы и т. д.).
    - Модуль включает в себя блок преобразования полученных документов в формат PDF на основе LibreOffice. В данном блоке преобразования осуществляется

запуск нескольких экземпляров LibreOffice в режиме сервиса, контроль работоспособности этих экземпляров, их перезапуск при необходимости.

Схема модулей информационной системы, построенной на платформе BIV.

C2. Container (модуль).
Модуль — высокоуровневый логический элемент архитектуры, состоящий из компонентов и представляющий из себя законченное функциональное решение.

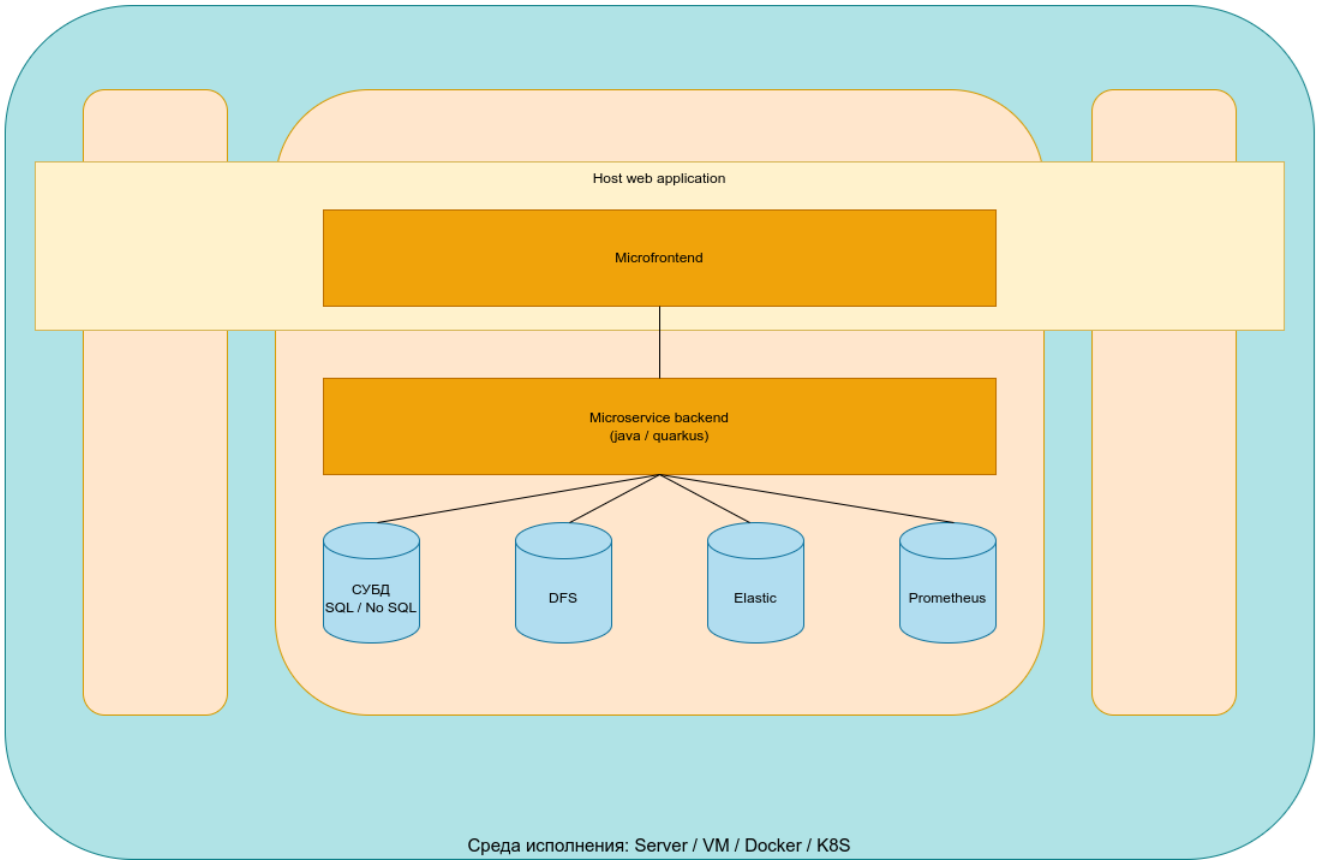


На схеме отражена функциональная структура (модуль отвечает за какую-то функцию).
На схеме не показаны элементы модулей (СУБД, файловые хранилища, балансировщики, веб-серверы и т.п.). Это уровень детализации отдельного модуля.

Схема прикладного модуля информационной системы, построенной на платформе BIV.

C3. Component.

Окружение прикладного сервиса



Состав прикладного модуля информационной системы, построенной на платформе BIV.

C3. Component.
Компонент — составная часть модуля, элемент, реализующий некоторую отдельную функциональность (как системную, так и прикладную).
Не является автономным законченным элементом архитектуры.

Обеспечивающие, базовые, прикладные серивсы могут быть построены с помощью компонент платформы.

Состав прикладного сервиса

