

Описание технической архитектуры системы

Оглавление

Список компонентов	2
Взаимодействие компонентов	3
Путь запроса ws/wss	4
Путь запроса http/https	4
Схема взаимодействия сервисов приложения	5
Технические требования клиента	6
Технические требования REST API	6

Список компонентов

Приложение состоит из следующих сервисов:

1. Клиент (React)
2. Websocket Service (Node.js)
3. Rest API Service (Laravel)
4. Redis
5. PostgreSQL
6. S3 - диск
7. Внешняя система IP телефонии
8. Nginx

Описание компонентов

1. Клиент реализован на языке javascript библиотеке React. Предоставляет интерфейс взаимодействия с пользователем
2. Websocket сервис реализован на языке javascript с использованием фреймворка node.js. Используется для отслеживания изменений сущностей в режиме реального времени.
3. Rest API сервис реализован на языке php версии 8.3 с использованием фреймворка Laravel 11. Используется для управления сущностями (создание, изменение, удаление в корзину, восстановление, удаление из базы данных) и получения данных сущностей, а также авторизации клиента
4. Redis - высокопроизводительное не реляционное хранилище данных типа «ключ-значение» с открытым исходным кодом. Используется как сервис кеширования данных.
5. PostgreSQL – свободная объектно-реляционная система управления базами данных (СУБД) с открытым исходным кодом. Используется как основная система хранения данных.
6. Внешняя система IP телефонии – это внешняя система, предоставляющая услуги связи IP телефонии, не зависящая от данного приложения и обеспечивающая услуги связи в режиме 24/7 с реализацией REST API для управления звонками, списками номеров и сотрудниками.
7. Nginx – это программное обеспечение с открытым исходным кодом, используется в качестве высокопроизводительного веб-сервера, обратного прокси-сервера и балансировщика нагрузки.

8. S3 – диск это объектное облачное хранилище, которое позволяет хранить практически неограниченное количество данных с доступом по уникальной URL-ссылке. Данные хранятся в бакетах — виртуальных контейнерах, где располагаются файлы (объекты). Каждый объект имеет уникальный ключ, который позволяет идентифицировать его внутри бакета, и сопровождается метаданными, описывающими его свойства, например, размер, тип или дату загрузки.

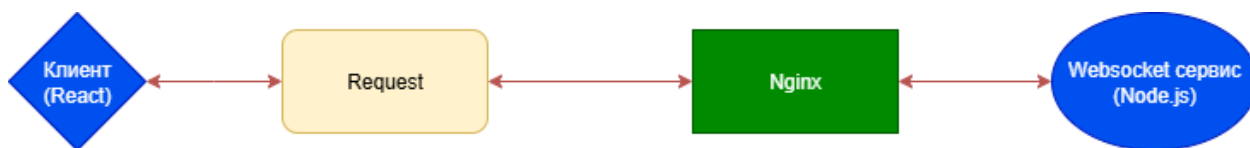
Взаимодействие компонентов

Клиент по протоколу http/https и ws/wss обращается к веб-серверу, веб-сервер проксирует запрос на сервис в зависимости от протокола.

При обращении клиента по протоколу ws/wss веб сервер проксирует запрос на websocket сервис и гарантирует постоянное соединение клиента и websocket сервиса для постоянного обмена информацией в режиме реального времени.

При обращении клиента по протоколу http/https веб-сервер проксирует запрос на демона php-fpm который в свою очередь, запускает REST API сервис. Запрос попадает в слой посредник проходит необходимые проверки, после чего попадает в валидатор запроса для обеспечения безопасности данных. Валидация данных гарантирует безопасную обработку запроса. Отвалидированные данные проходят далее в контроллер, определенный в списке маршрутов приложения в зависимости от метода запроса и URL адреса, из которого, в виде ассоциативного массива, проходят в сервисный слой для дальнейшей обработки. Сервисный слой обеспечивает всю бизнес-логику приложения включая взаимодействие с Redis, PostgreSQL, S3-диском, внешними системами, асинхронную обработку запросов и обновление данных в каналах websocket сервиса. REST API сервис взаимодействует с Redis, PostgreSQL по средствам встроенных в ядро php драйверов, которые обеспечивают полный доступ к данным и взаимодействуют с серверами как на локальной так и на удаленной машине обеспечивая удобный интерфейс выполнения запросов по протоколу tcp/ip или unix-socket. Взаимодействие с S3 – диском обеспечено посредством протокола взаимодействия S3, разработанного компанией Amazon, а также REST API данного сервиса и http/https протоколу.

Путь запроса ws/wss



Путь запроса http/https

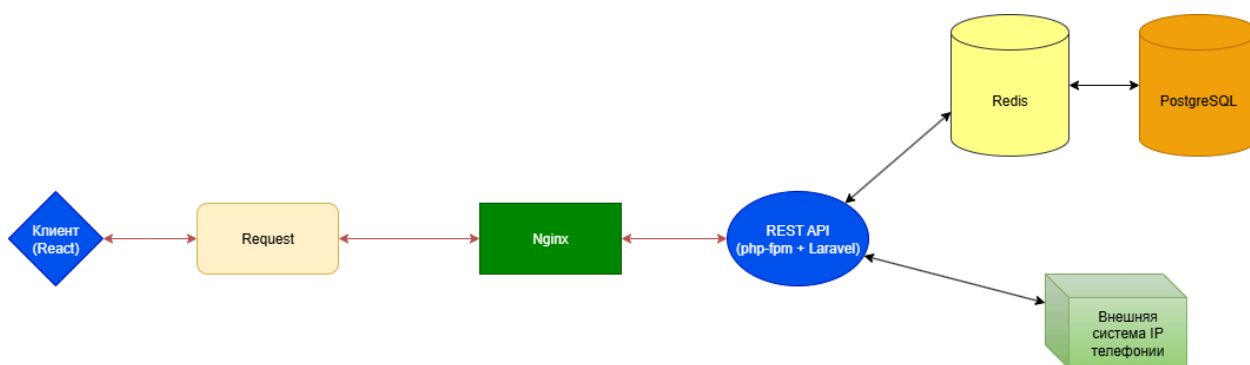
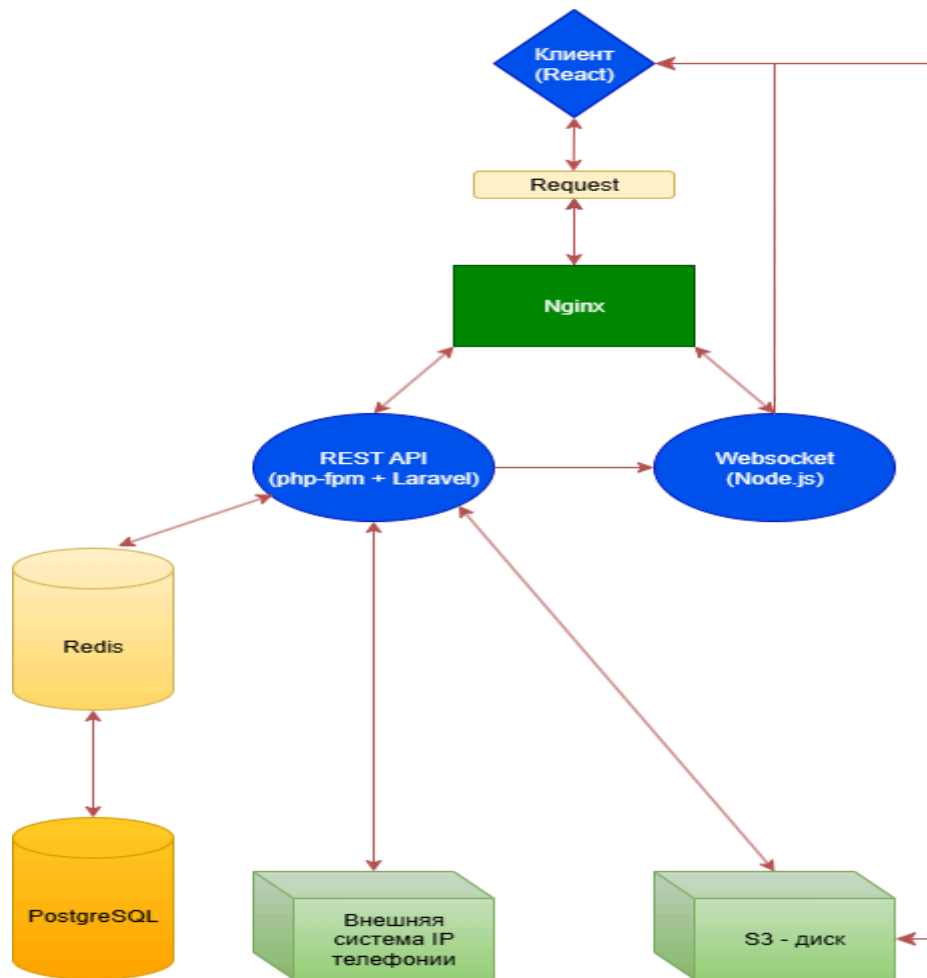


Схема взаимодействия сервисов приложения



Сотрудник по средствам веб-интерфейса взаимодействует с клиентским слоем. Клиентский слой взаимодействует с серверным слоем по протоколам http/https и ws/wss. Серверный слой взаимодействует с сервером Redis с помощью встроенного в ядро php драйвера, для получения кешированных данных, в случае их отсутствия данные поступают из БД PostgreSQL с помощью встроенного в ядро php драйвера PDO и кешируются в Redis для снижения нагрузки на БД. С внешними системами и websocket сервисом, серверный слой взаимодействует по протоколу http/https посредством REST API технологии.

Технические требования клиента

- Актуальная версия web-браузера Google Chrome
- ОС любая поддерживающая актуальную версию необходимого браузера

Технические требования REST API

- ОС Ubuntu 22.04 и выше
- php-fpm 8.3 и выше
- Наличие сервиса Redis 5.6 или выше для кеширования данных
- Наличие реляционной базы данных PostgreSQL 16.0 или выше для хранения данных
- Наличие внешнего сервиса IP телефонии
- Наличие внешнего S3 – диска для хранения статических объектов
- Наличие установленной в ОС библиотеки ffmpeg – для определения длительности записей разговоров