Описание технической архитектуры программного обеспечения

Программное обеспечение построено по модульному принципу. Это позволяет развивать модули параллельно, уменьшить связанность системы и увеличить ее стабильность.

Выделены следующие модули:

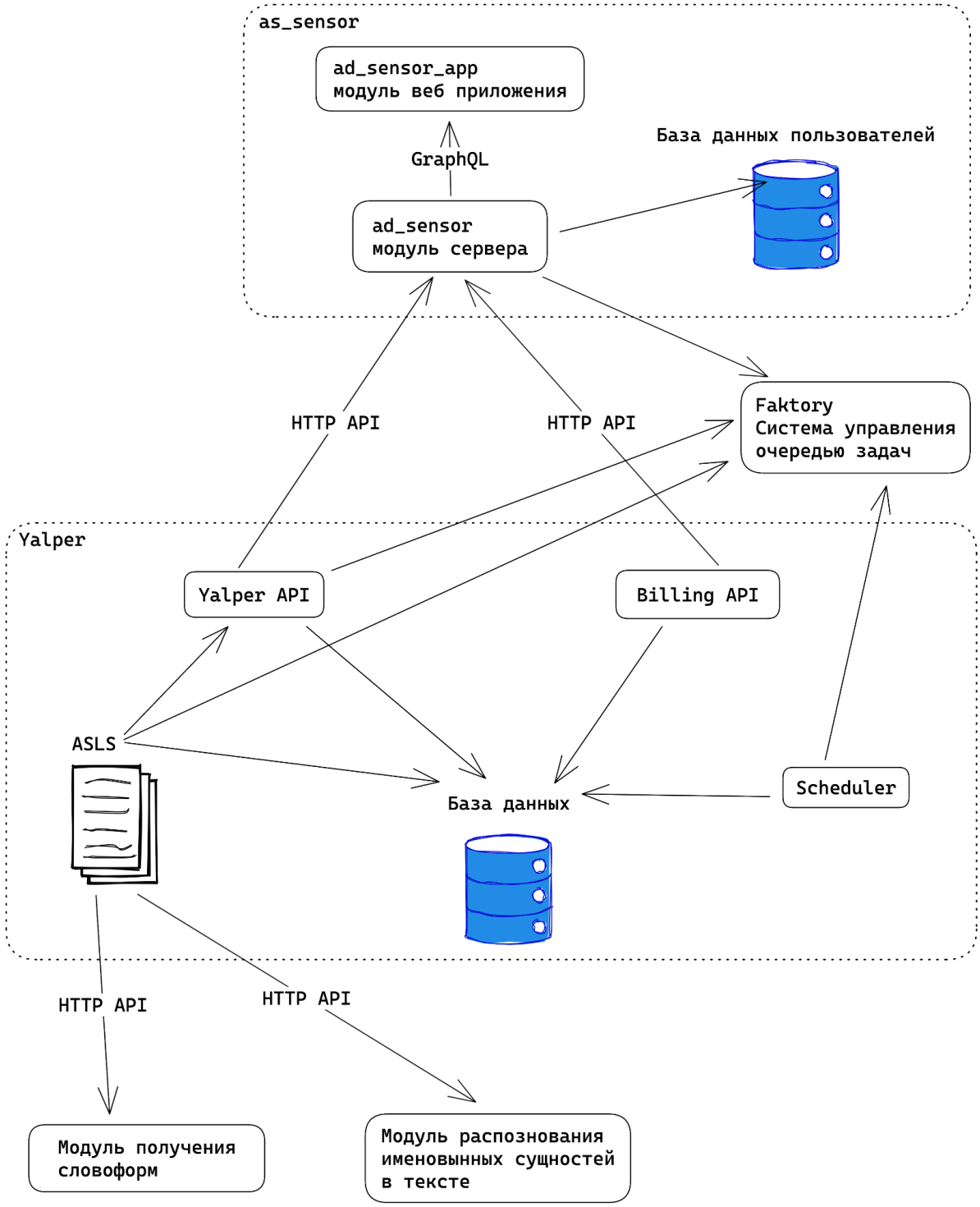
- Модуль выполняющий проверку рекламных кабинетов на ошибки (Yalper);
- Модуль отображения данных, реализующий веб интерфейс для взаимодействия с пользователями (ad_sensor);
- Модуль получения словоформ;
- Модуль распознавания именованных сущностей в тексте.

Для взаимодействия между модулями используется REST API. Формат передаваемых данных JSON.

В качестве подсистемы хранения и управления задачами было выбрано open-source решение Faktory. Оно позволяет с минимальными настройками получить единую очередь задач с клиентами, написанными на разных языках. Она не требует глубокой изначальной настройки и проста в установке и администрировании.

В качестве базы данных используется Postgresql.

Схема модулей и их взаимодействия



Модуль веб-интерфейса «ad_sensor»

Модуль состоит из следующих частей:

- Сервера;
- Веб приложения;
- Базы данных.

Для «общения» между сервером и веб приложением используется протокол GraphQL.

Веб приложение реализовано по принципу одностраничного приложения (SPA) и при первом открытии загружает в веб-браузер пользователя всё что необходимо для отображения сайта. Это позволяет уменьшить время открытия страниц (отзывчивость приложения). Веб-приложение реализовано на фреймворке «React» с использованием UI-фреймворка «MaterialUI».

Сервер реализован на языке «Elixir» с использованием фреймворка «Phoenix» и библиотеки «Absinth» для реализации GraphQL API.

Внутри сервер построен с использованием принципа контекстов для лучшего структурирования доменных областей и выделение интерфейсов их общения.

Для всех модулей обязательно написание unit-тестов. На GraphQL API обязательно написание интеграционных тестов.

Внутрь сервера встроен модуль, реализующий обработку задач из очереди Faktory.

В качестве базы данных используется PostgreSQL.

Модуль проверки рекламных кабинетов «Yalper»

Модуль состоит из следующих частей:

- Внешний API;
- API-биллинг;
- Сервис обработки задач;
- Планировщик задач.

Все части реализованы на языке Golang и хранятся в одном монорепозитории.

Все сервисы общаются между собой через HTTP API.

Планировщик задач отвечает за постановку задач в очередь по заранее заданному расписанию. Он осуществляет “умную” постановку задач: знает о всех подключенных доступах в рекламные кабинеты пользователей и доступных проверках и ставить задачи на каждую проверку на каждого пользователя.

Сервис обработки задач отвечает за обработку всех задач. Он масштабируется за счет запуска новых экземпляров сервиса.

При обнаружении ошибок в выполняемых задачах они попадают с отдельную очередь. Для предотвращения падения инстансов (экземпляров) сервиса, в нем реализован механизм, который прекращает принимать задачи на конкретный инстанс (экземпляр) при достижении пределов свободной памяти на машине.

Каждый инстанс (экземпляр) сервиса обработки задач может отдельно конфигурировать:

- задачи из каких очередей он выполняет;
- сколько одновременно задач может выполняться;
- процент свободной памяти.

Модуль получения словоформ

Модуль реализован на языке Python и имеет интерфейс HTTP API.

Поставляется в виде образа docker. Он развернут в виде отдельного инстанса (экземпляра).

Модуль распознавания именованных сущностей в тексте

Используется open-source решение DeepPavlov. Оно поставляется в виде готового docker-образа и имеет HTTP API.