Новосибирский Государственный Университет Факультет Информационных Технологий

Техническое описание проекта по курсу ООАД

Comics search service

Студенты ФИТ НГУ

Пучков Даниил Дмитриевич

Богаченков Иван Андреевич

Группа 23203

Версия 1.0.0

Содержание

Оглавление

Техническое описание проекта по курсу ООАД

1. Введение

1.1. Цель

Данный документ представляет собой техническое описание проекта **Comics search service** и содержит основные требования к разрабатываемой в рамках проекта программной системе и описание архитектуры программного решения.

1.2. Область действия

Документ разработан в рамках проекта **ComicsSearchService** на основе стандартного шаблона и предназначен для использования студентами ФИТ и преподавателями дисциплины ООАД.

1.3. Определения и сокращения

Telegram бот - это роботизированный аккаунт в мессенджере Telegram, который запрограммирован на автоматическое совершение действий

1.4. Краткое описание

Содержание данного документа построено таким образом, чтобы дать ответ на следующие вопросы:

- Какие проблемы предметной области должен решать будущий программный продукт
- Посредством какой функциональности системы будут достигнуто решение проблем предметной
- области
- Какова архитектура программного решения

Описание предметной области и проблем, для решения которых предназначен будущий программный продукт, приведены в разделе 2.

Раздел 3 содержит описание требований к программному решению, раздел 4 – описание архитектуры выбранного решения.

2. Предметная область проекта

Предметной областью является информационный поиск по архиву веб-комиксов XKCD.

Каждый комикс представляет собой единицу контента, содержащую изображение и текстовые метаданные: заголовок (title), альтернативный текст (alt) и текстовую расшифровку диалогов (transcript). Эти данные доступны публично, но распределены по отдельным JSON-ресурсам источника.

2.1. Существующие проблемы

1. Отсутствие структурированного поиска: Оригинальный источник предоставляет доступ по ID комикса, но не имеет встроенного API для глубокого полнотекстового поиска по содержимому диалогов или alt-текстов.

2. Информационный шум: "Сырой" текст содержит знаки препинания, стоп-слова (предлоги, артикли) и слова в разных грамматических формах, что затрудняет точное нахождение контента по ключевым словам.

3. Латентность и зависимость: Прямой перебор данных на источнике при каждом запросе невозможен из-за сетевых задержек и ограничений нагрузки на внешний сервер.

2.2. Предполагаемое решение

Разработанная **распределенная поисковая система (Distributed Search Engine)**, которая решает эти проблемы:

1. **Агрегация и Хранение:** автоматически скачивает и синхронизирует локальную базу данных (PostgreSQL) с источником, обеспечивая автономность и быстрый доступ к данным.
2. **Лингвистическая нормализация:** очищает текст от шума и приводит слова к их основе (стемминг), что позволяет находить "running", вводя "run".
3. **Высокопроизводительный поиск:** реализует **инвертированный индекс (Inverted Index)** в оперативной памяти (или поиск по массивам в БД), обеспечивая мгновенный отклик на поисковые запросы.
4. **Единая точка входа:** предоставляет защищенный и удобный интерфейс для клиентов, скрывая сложность внутренней реализации.

3. Требования к программному решению

Данный раздел описывает требования к программной системе, разрабатываемой в рамках проекта ComicsSearchSerice

3.1. Роли

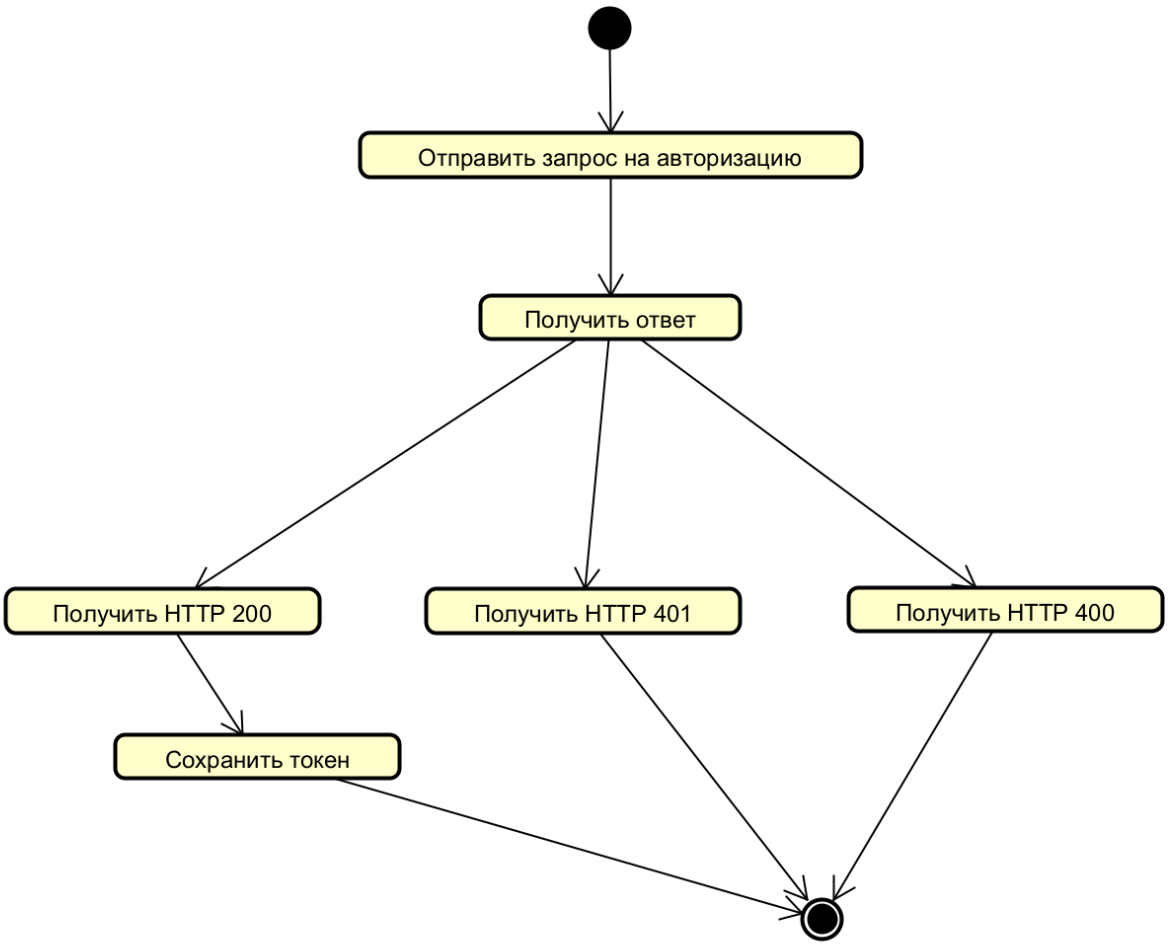
Роль — это что-то (например: другая система) или кто-то (например: человек) вне системы, которые взаимодействуют с ней. В предлагаемой к разработке системе идентифицированы следующие роли:

1. Гость - пользователь, который не осуществил аутентификацию в сервисе. Может пользоваться только поиском комиксов.

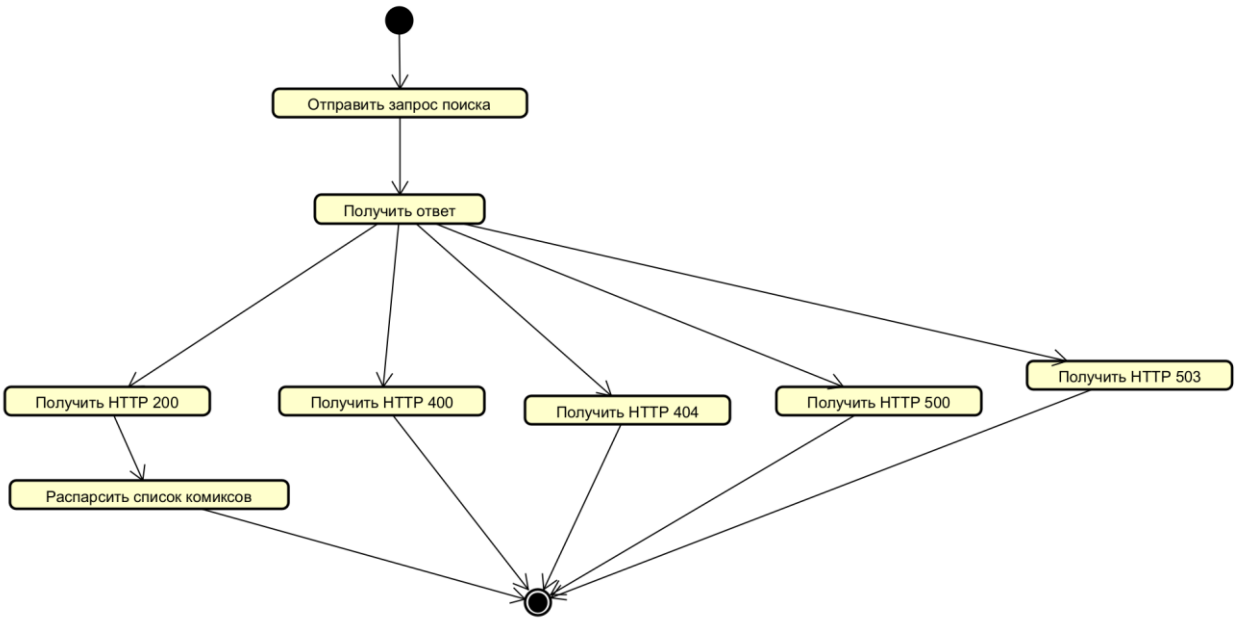
2. Администратор - пользователь, имеющий доступ ко всем предоставляемым возможностям API, таким как просмотр статистики БД, просмотр статуса воркера, запуск обновления БД, очищение БД.

3.2. Функциональные требования для роли Гость

3.2.1. Авторизация

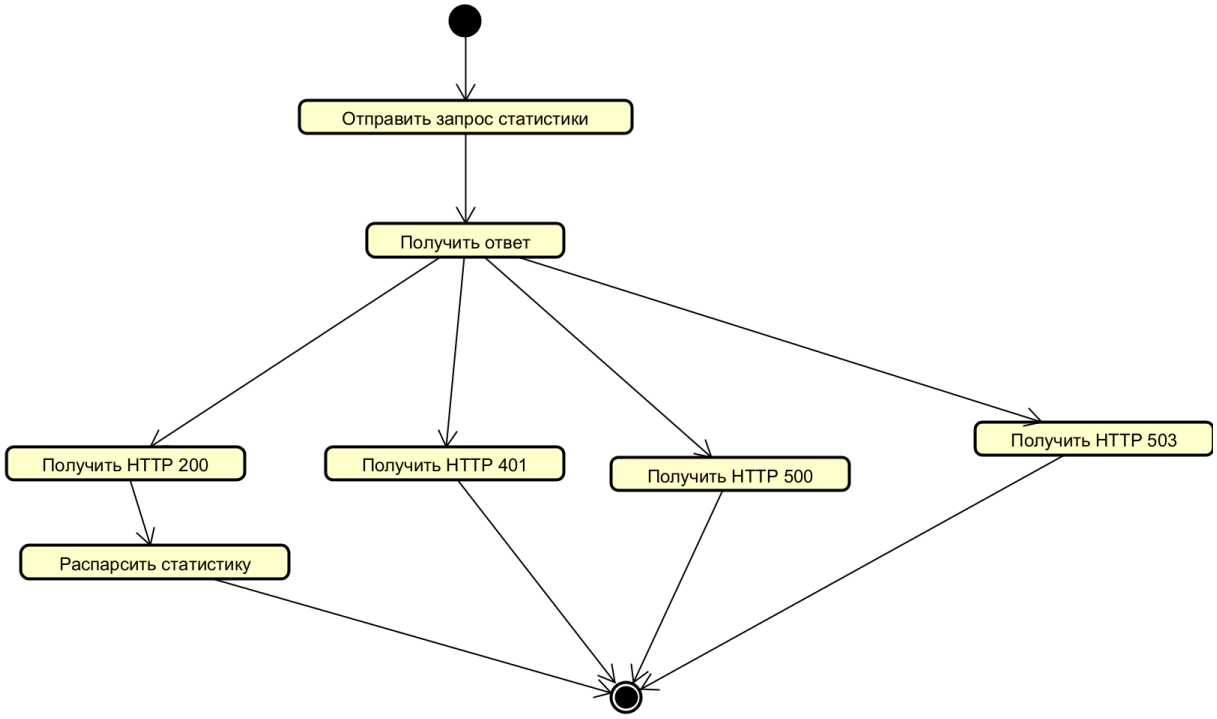


3.2.2 Поиск комиксов

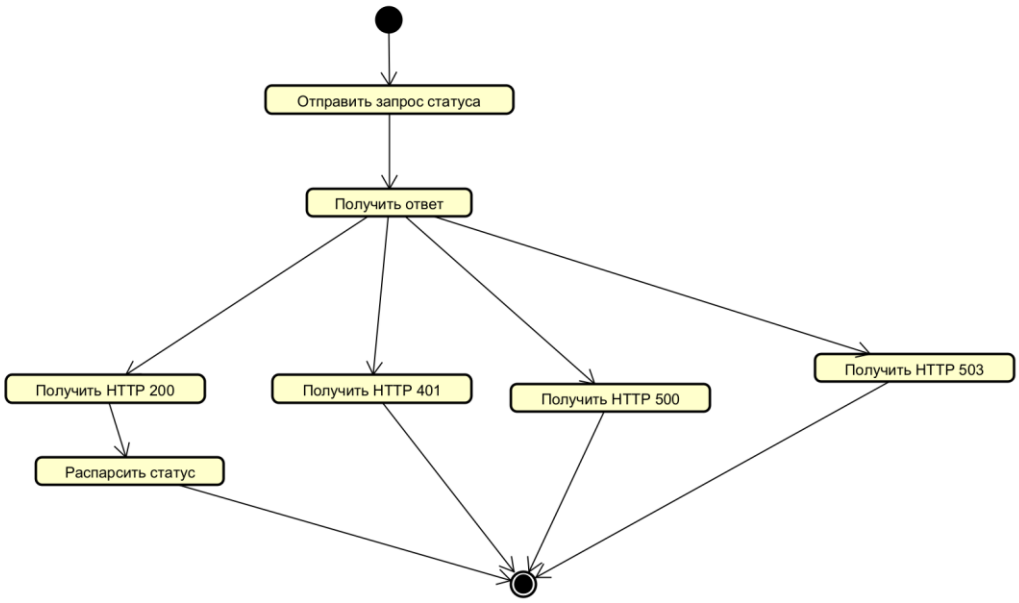


3.3. Функциональные требования для роли Администратор

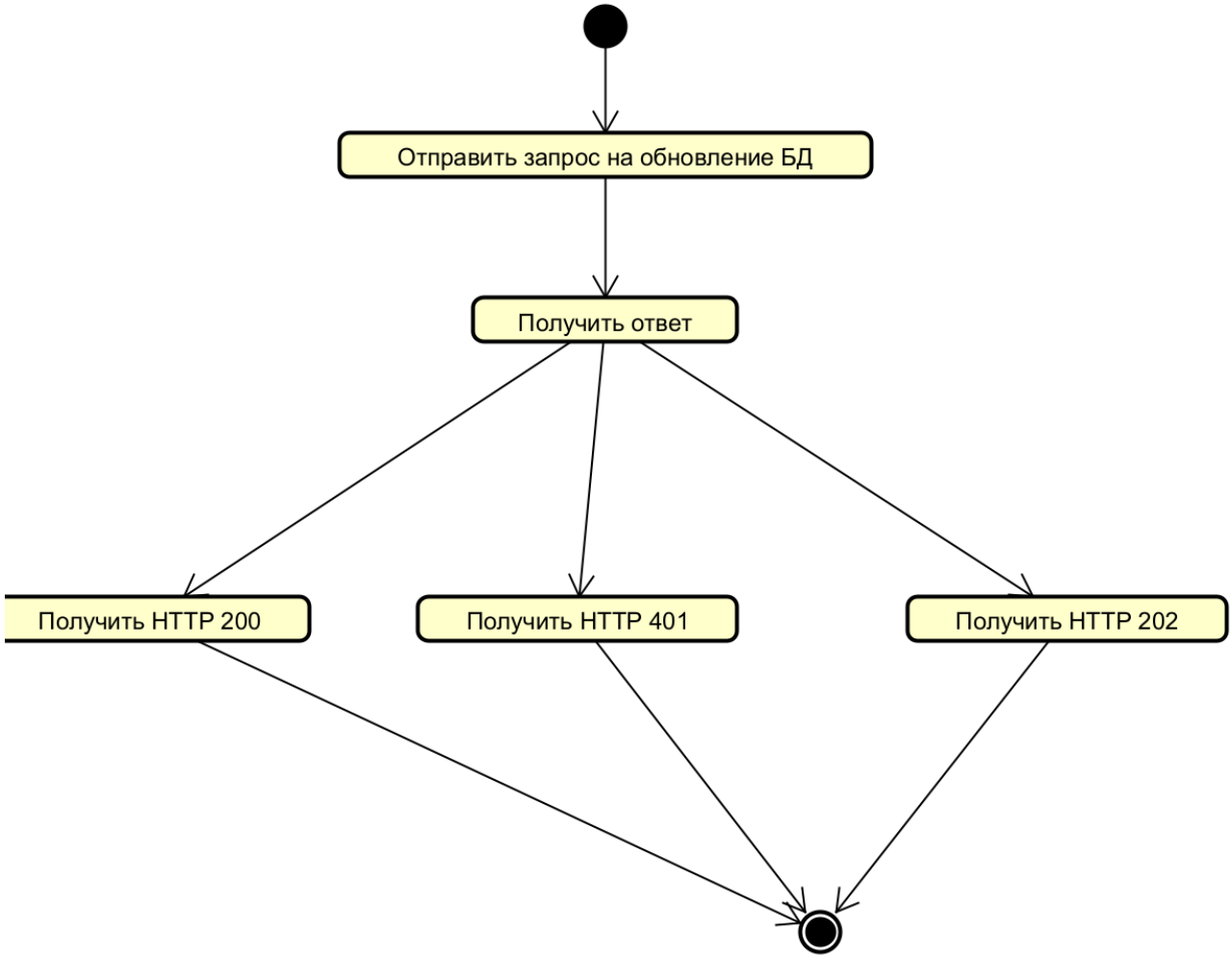
3.3.1. Просмотр статистики БД



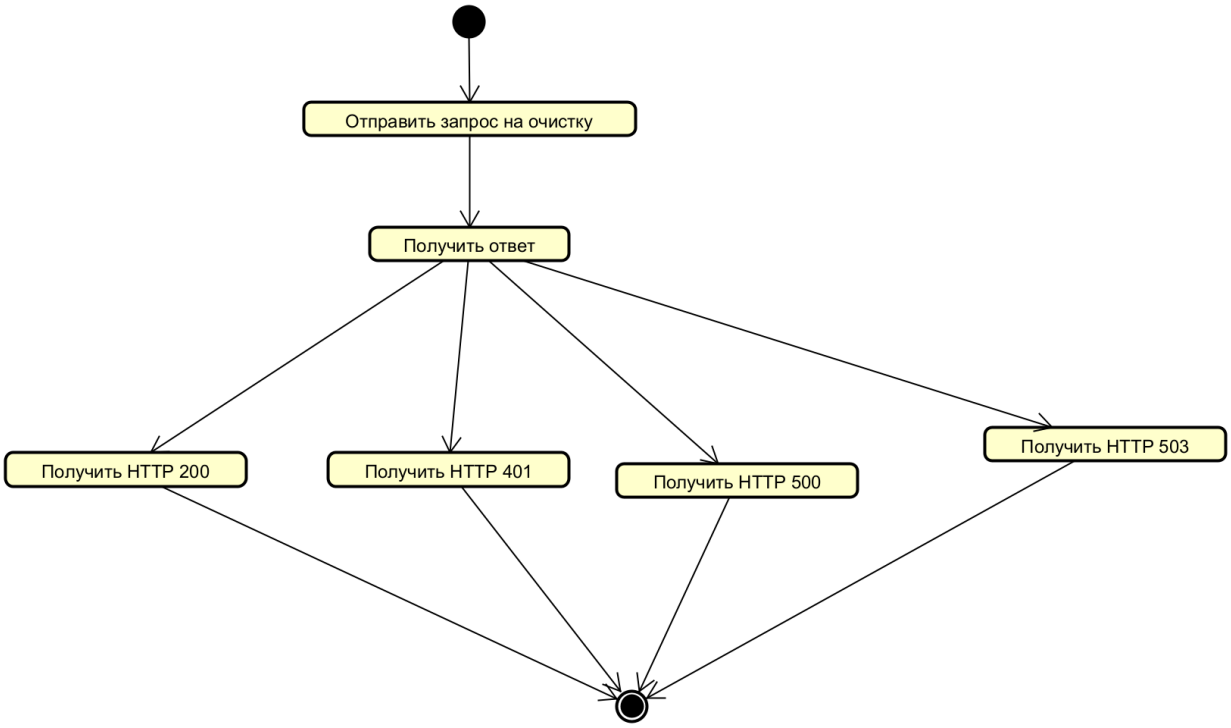
3.3.2. Просмотр статуса воркера



3.3.3. Запуск обновления БД


Отправить запрос на обновление БД
Получить ответ
Получить HTTP 200
Получить HTTP 401
Получить HTTP 202

3.3.4. Очистить БД



3.4. Нефункциональные требования

3.4.1. Производительность

- При любых действиях при прогретом индексе, сервис должен иметь быстрое время ответа, не превышающее 3 секунд, чтобы обеспечить хороший пользовательский опыт.
- Система должна поддерживать обработку как минимум 20 запросов в секунду.

3.4.2. Надежность

- Сервис должен быть доступен более 99% времени в сутки

3.4.3. Безопасность

- Система должна иметь уровни доступа к API.

3.4.4. Масштабируемость

- Система должна быть масштабируемой

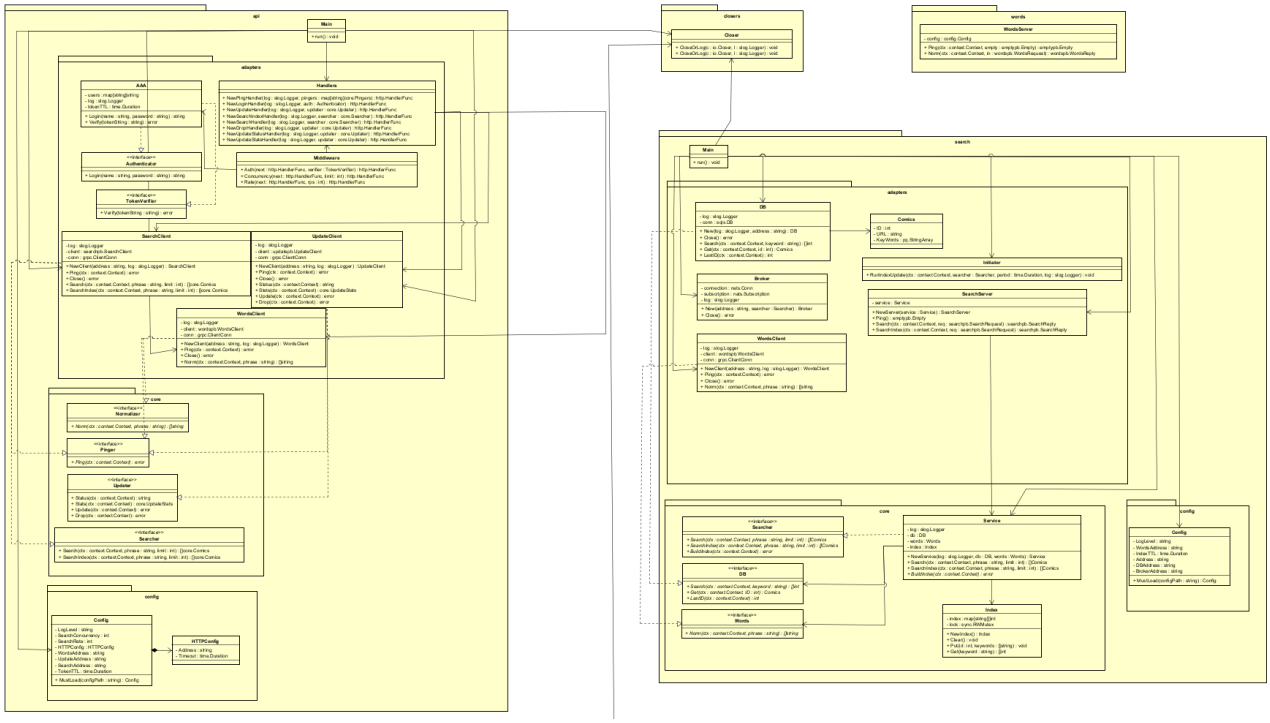
3.4.5. Документация

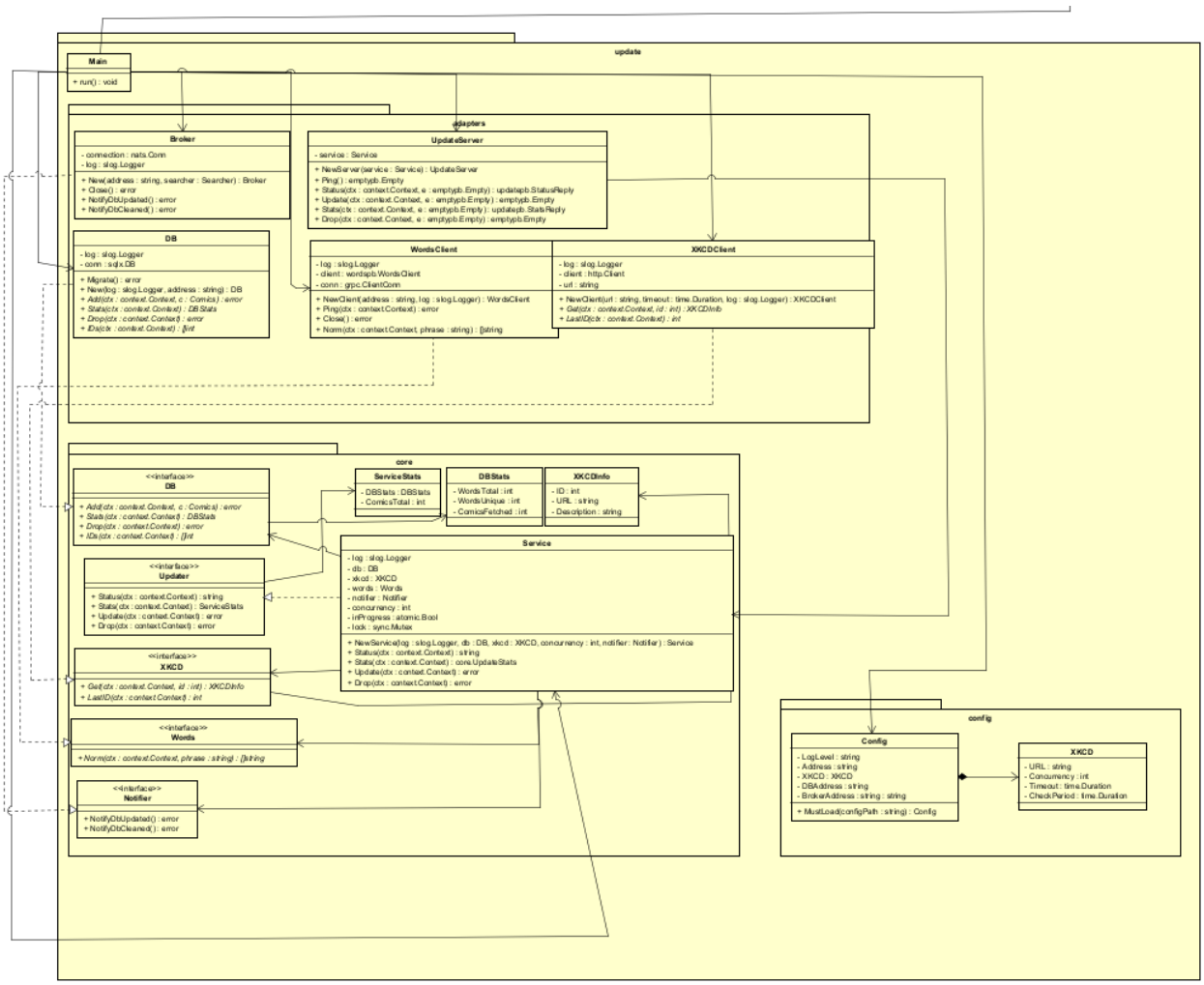
- Код должен быть снабжен комментариями

4. Обзор архитектуры

Этот раздел описывает архитектуру системы.

4.1.1. Компонентная модель системы





4.1.1.1. adapters

В модуле располагаются классы, являющиеся адаптерами для портов в core.

4.1.1.2. core

В модуле располагаются порты, основные классы и бизнес логика микросервиса.

4.1.1.3. config

В модуле располагаются классы для настройки сервиса по конфиг файлам.

4.1.1.4. closers

В модуле располагаются классы, отвечающие за удобное закрытие ресурса и логирование.

4.1.1.5. api

В модуле располагаются классы, использующиеся для работы API Gateway сервиса.

4.1.1.6 update

В модуле располагаются классы, использующиеся для работы update микросервиса.

4.1.1.7 search

В модуле располагаются классы, использующиеся для работы search микросервиса.

4.1.1.8 words

В модуле располагаются классы, использующиеся для работы words микросервиса.

4.1.2. Компоненты сторонних производителей

Библиотеки:

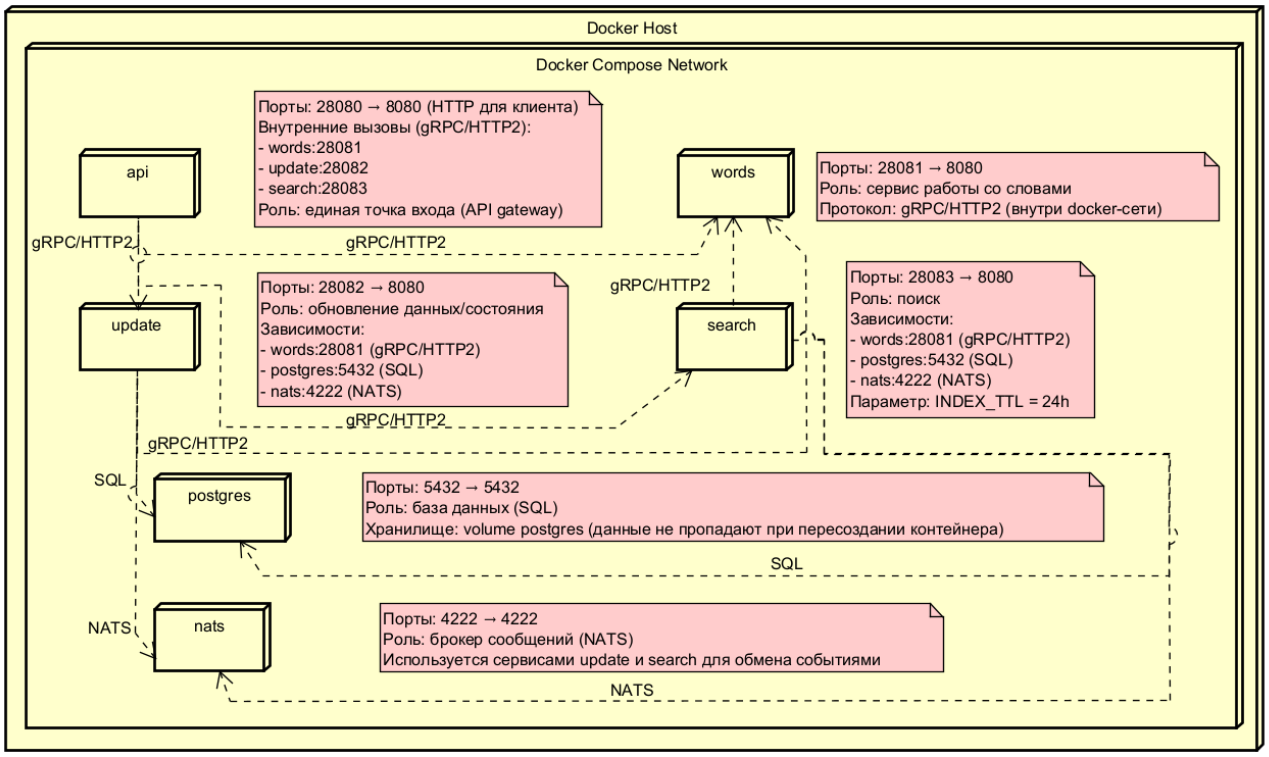
- github.com/lib/pq - Драйвер PostgreSQL
- google.golang.org/grpc - gRPC фреймворк
- google.golang.org/protobuf - Сериализация данных
- github.com/golang-jwt/jwt/ - Работа с JWT
- github.com/golang-migrate/migrate/v4 - Миграции БД
- github.com/ilyakaznacheev/cleanenv - Чтение конфигурации
- github.com/jackc/pgx/v5 - Драйвер PostgreSQL
- github.com/jmoiron/sqlx - Расширение SQL
- github.com/kljensen/snowball - Стемминг текста
- github.com/nats-io/nats.go - Клиент NATS

Программное обеспечение:

- postgresql - база данных
- nats – брокер сообщений
- Docker - контейнеризация приложения
- Docker compose

4.1.3. Схема развертывания приложения

Для развертывания приложения используются Docker контейнеры. Для запуска контейнеров используется Docker-compose.



Для развертывания приложения необходим сервер/пк с минимальными требованиями:

• Жесткий диск 20 гб

• Оперативная память 1гб

• Процессор с частотой 2.4 ггц

• Установленное программное обеспечение docker, docker-compose

• Подключение к интернету

5. Допущения и ограничения

Ограничения:

• На разработку диаграмм (use-case, sequence, классов, пакетов) было применено временное ограничение в 1 месяц

• На разработку приложение было применено временное ограничение в 1.5 месяца

Допущения:

• При разработке проекта принято допущение, что число обращений к боту в единицу времени значительно (более чем в 10 раз) снижается в ночное время, что позволяет в период производить обновление программного обеспечения

системы, требующее полной перегрузки и остановки сервиса на период до 5 минут

6. Известные проблемы

Ниже приводятся известные на данный момент проблемы и недоработки выработанного программного решения, а также возможные пути их устранения в последующих итерациях проекта.

6.1. Зависимость от работоспособности брокера сообщений NATS

- **Проблема:** Архитектура сервисов полагается на постоянную доступность NATS для доставки событий обновления индекса.
- **Ранг:** Высокий
- **Влияние на проект:** При отказе брокера нарушается синхронизация данных между сервисами, что приводит к неактуальной поисковой выдаче или полной остановке процессов обновления.
- **Пути решения:** Настройка кластера NATS (High Availability), реализация механизмов повторной отправки сообщений (Retry Policies) и локальных очередей (Dead Letter Queues) на стороне сервисов

6.2 Ограничение объема индекса оперативной памятью (In-memory index)

- **Проблема:** Размер поискового индекса и словаря жестко ограничен доступным объемом оперативной памяти (RAM) узла.
- **Ранг:** Высокий
- **Влияние на проект:** Риск аварийного завершения работы приложения (OOM) при росте объема данных. Невозможность индексации большого количества комиксов без вертикального масштабирования (увеличения памяти сервера).
- **Пути решения:** Реализация механизма сброса индекса на диск (fallback to DB/Disk), использование специализированных поисковых движков (Elasticsearch/Sphinx) или шардирование индекса по нескольким узлам.

6.3 Отсутствие поддержки многоязычности

- **Проблема:** Алгоритмы нормализации текста и стемминга оптимизированы исключительно под английский язык.
- **Ранг:** Средний
- **Влияние на проект:** Некорректная обработка запросов на других языках, невозможность расширения аудитории за пределы англоязычного сегмента.
- **Пути решения:** Подключение библиотек для морфологического анализа других языков, внедрение слоя перевода интерфейса (i18n).