ООО «КВАНТУМ АРТ»

115184, Москва, Озерковский переулок, д. 12
тел. (495) 783-65-74

# Компонент QP8.Search.Integration продукта QP8.Search

Документация разработчика

Москва
2023

## Назначение документа

Настоящий документ – руководство к компоненту QP8.Search.Integration продукта QP8.Search. Документ предназначен для разработчиков, реализующих модули индексации данных, хранящихся в продукте QP8.CMS.

## История изменений

| Версия | Дата | Автор | Описание |
|---|---|---|---|
| 1.0 | 26.01.2023 | Григорьева М.А. | Создание документа |
| | | | |
| | | | |
| | | | |

# Оглавление

# 1. Общие сведения

Компонент QP8.Search.Integration предназначен для индексации данных QP8.CMS в Elasticsearch для последующего поиска по ним с помощью QP8.Search.API.

Компонент состоит из двух частей:

- базовая часть, отвечающая за основной функционал индексации;
- подключаемые модули (плагины), содержащие данные по индексации информационного содержания, специфичную для каждого проекта.

Для интеграции с компонентом QP8.Search.Integration требуется реализовать только подключаемый модуль, в базовую часть компонента вносить изменения не требуется.

# 2. Архитектура

Процесс индексации затрагивает три компонента:

- QP8.CMS (база данных, в которой хранятся данные);
- QP8.Search.Integration, отвечающий за процесс индексации;
- Elasticsearch/OpenSearch, выступающий в роли подсистемы хранения данных.

Процесс индексации базируется на заданиях, которые запускаются согласно заданному в настройках расписанию (подробнее о настройках расписания см. в разделе «Настройка приложения»).

Компонент QP8.Search.Integration согласно расписанию, заданному в конфигурации, запускает процесс индексации. Индексация запускает итерационное чтение данных из базы данных, в которой QP8.CMS хранит информацию, выбирая данные в количестве, указанном в настройках компонента. После выборки данные проходят этап преобразования в формат, подходящий для хранения и индексации средствами Elasticsearch/OpenSearch, например, HTML-документ освобождается от тегов, формируются правильные ссылки на страницы сайта, на которых размещены индексируемые данные и выполняются любые другие действия, реализованные разработчиком в подключаемом модуле индексации. После чего компонент индексации отправляет данные в Elasticsearch/OpenSearch, который, в свою очередь, помимо сохранения, индексирует их на основе заранее определенного шаблона индексации.

# 3. Отказоустойчивая индексация

При индексации данных происходит создание нового индекса и удаление старого. Для обеспечения бесперебойного функционирования API поиска и минимизации рисков отказа при неудачной индексации применяется логика работы с Alias в Elasticsearch/OpenSearch.

На каждый индекс назначается Alias, составляемый по имени проекта и по имени объекта индексации. Название индекса, в свою очередь, помимо названия проекта и объекта индексации содержит временной отпечаток, присваиваемый ему в момент создания.

Все обращения из API поиска осуществляются по имени Alias, а не по имени индекса, что обеспечивает бесперебойное функционирование API поиска даже во время выполнения полной индексации данных.

Процесс индексации выстроен таким образом, что переключение Alias на новый индекс происходит только после успешной индексации, что исключает возможность работы API поиска по ошибочно или неполностью сформированному индексу.

# 4. Виды индексации

Индексация в компоненте QP8.Search.Integration делится на три вида:

- полная индексация;
- частичная индексация;
- индексация ролей доступа.

## 4.1. Полная индексация

Полная индексация запускается согласно расписанию, установленному в конфигурации.

При запуске выполняется следующая последовательность действий:

- из базы данных, в которой QP8.CMS хранит данные, загружаются все данные контентов, описанные в подключаемом модуле;
- нормализация данных (очистка от лишних данных, формирование адресов и прочие модификации);
- отправка данных в Elasticsearch/OpenSearch для индексации и сохранения, сохранение происходит в новый индекс, который создаётся в моменте загрузки данных в Elasticsearch/OpenSearch;
- переключение Alias на новый индекс;
- удаление старого индекса.

Стоит понимать, что полная индексация создаёт нагрузку как на базу данных, так и на Elasticsearch/OpenSearch. При конфигурировании расписания полной индексации стоит задавать частоту полной индексации как можно реже.

## 4.2. Частичная индексация

При частичной индексации, в отличии от полной, происходит не выгрузка всех данных из источника данных, а загрузка изменений, сделанных в источнике данных за последние два дня (текущий день и предыдущий). Стоит понимать, что это не 48 часов, а **с 00:00:00 вчерашнего дня относительно момента запуска частичной индексации.**

В этом режиме индексации, в отличии от полной индексации, работа осуществляется с данными в рабочем индексе, создание нового индекса и переключение Alias не происходит.

Данные, которые были изменены, обновляются во время частичной индексации. Данные, которые были добавлены, создаются в рабочем индексе.

За счёт не полного чтения всех возможных данных, а выборки только части данных, этот вид индексации создаёт меньшую нагрузку на БД и Elasticsearch/OpenSearch, следовательно, запускать этот вид индексации можно чаще полной, но не реже чем раз в два дня.

При выполнении частичной индексации выполняется следующая последовательность действий:

- из базы данных, в которой QP8.CMS хранит данные, загружаются данные, изменённые за последние 2 дня;
- нормализация данных (очистка от лишних данных, формирование адресов и прочие модификации);
- отправка данных в Elasticsearch/OpenSearch для индексации и сохранения.

## 4.3. Индексация ролей доступа

Если в продукте QP8.CMS используется ролевая модель доступа пользователей к информации, размещённой на сайте под управлением QP8.CMS, то можно индексировать данные с учётом ролей доступа.

Глобально данные разделяются на два вида:

- доступные всем (публичные);
- доступные пользователям с определённой ролью (приватные).

Для публичных данных используется название роли, определённое в конфигурации, например, QP8.Search.Reader. Важно помнить, что указанное в конфигурации имя публичной роли должно отличаться от любых имён ролей, заведённых в QP8.CMS, а также должно совпадать с тем же именем, указанным в конфигурации компонента QP8.Search.API.

При индексации ролей возможны три режима работы с ролями:

- ограничение доступа ко всем данным (проявляется в виде ограничения доступа к индексу контента при поиске);
- ограничение доступа к каждой отдельной статье (реализуется дополнительным полем в каждом индексе, содержащим список ролей, которым разрешено выбирать каждый отдельный документ индекса);
- комбинированный режим, включающий вышеописанные режимы.

### 4.3.1. Ограничение по всему контенту

Если в QP8.CMS ограничение применяется на уровне всего контента, а не отдельных статей контента, то стоит использовать этот режим работы.

В этом случае согласно заданному в конфигурации интервалу, запускается фоновый обработчик, который проверяет все объекты индексации, для которых необходимо проверять ролевые ограничения. Для каждого объекта обработчик пытается получить список ролей, которым разрешено читать из контента. Если список ролей пустой (ограничений нет), то обработчик рекурсивно проходит по родителям контента до самого верхнего уровня. Если у одного из родительских контентов указаны ограничения, то эти ограничения наследуются и для индексируемого контента. Если ограничения не найдены, то контенту добавляется роль по умолчанию (публичная роль).

В Elasticsearch/OpenSearch создаётся индекс ролей, куда пишутся соответствие индекса контента и массива ролей, которым разрешено искать в этом индексе.

При использовании API поиска с включенной поддержкой ролей, поиск будет сначала выбирать индексы, в которых разрешено искать роли, полученной в запросе. Далее API модифицирует запрос для поиска только в разрешенных ролях и только потом отправит запрос в Elasticsearch/OpenSearch. Подробнее про это можно прочитать в документации к компоненту QP8.Search.API.

### 4.3.2. Ограничение по статье контента

Если в QP8.CMS ограничение накладывается не на контент целиком, а на статью контента, то следует использовать этот режим работы.

В этом режиме фоновый индексатор ролей не запускается, а ограничения накладываются во время индексации контентов для каждой загруженной записи через вызов получения списка ролей вспомогательной библиотеки индексации ролей.

При получении ограничения ролей для статьи контента метод библиотеки индексации ролей сначала пытается получить роли для конкретной статьи контента, а если их не находит, то проверяет ограничение ролей всего контента по той же логике, что используется при ограничении по всему контенту.

После получения списка ролей их требуется сохранить в созданное для этого поле индекса.

При включенной поддержке ролевой модели поиск через API можно ограничить, задав роль пользователя, который осуществляет поисковый запрос. Для этого в параметре «$where» запроса в качестве ключа указывается

имя поля, в которое индексируются роли, а в качестве значения – роль пользователя. Подробнее – см. документацию по QP8.Search.API.

# 5. Ограничения одновременной индексации

Для уменьшения нагрузки на источник данных и на Elasticsearch/OpenSearch реализован механизм защиты от одновременного запуска нескольких индексацией.

В случае, если запущена индексация и в это же время пытается запуститься другая индексация, то сработает механизм защиты, которые выполнит одно из следующий действий:

- ожидание завершения уже запущенной индексации с последующим запуском ожидающей индексации;
- пропуск ожидающей индексации.

Решение принимается на основе типа индексации. Если в ожидание попадает полная индексация, то приложение идёт по первому сценарию, ждёт завершения уже запущенной индексации, после чего запускает полную. Это сделано с целью уменьшения рисков бизнеса из-за неактуальных данных в поисковой системе. Полная индексация запускается достаточно редко и является операцией высокой важности, её пропуск несёт в себе высокие риски.

Остальные индексации (частичная, роли или любые другие индексации, реализованные разработчиком в рамках проекта) несут в себе гораздо меньше рисков и могут запускаться куда чаще полной, в связи с чем они пропускаются и будут запущены повторно согласно своему расписанию.

Стоит помнить, что запустить любую индексацию вне очереди можно вручную в любой момент времени через компонент QP8.Search.Admin.

# 6. Автоматическая очистка документа

В ядре компонента реализован механизм автоматической очистки документа от HTML-разметки, удаляющий любые компоненты HTML и CSS из текстовых полей документа перед отправкой документов в Elasticsearch/OpenSearch.

При необходимости дополнительной модификации документа перед отправкой можно в рамках проекта собственной интеграции реализовать собственный обработчик и использовать его там, где это требуется.

# 7. Реализация собственной интеграции

Готовый вариант собственной интеграции можно увидеть на примере интеграции с порталом Demosite.

## 7.1. Создание проекта

Для реализации подключаемого модуля собственной интеграции необходимо загрузить исходный код проекта QA.Search, открыть его в предпочитаемой IDE и создать новый проект с типом «библиотека».

В проект в качестве зависимостей нужно подключить проекты QA.Search.Generic.DAL и QA.Search.Generic.Integration.QP.

Если планируется использовать ролевую модель пользователей из QP8.CMS, то так же нужно подключить проект QA.Search.Generic.Integration.Permissions.

## 7.2. Создание контекста работы с источником данных

Требуется создать класс контекста, который будет содержать в себе логику привязки объектов источника данных (БД) к объектам кода (моделям).

Класс контекста должен наследоваться от класса GenericDataContext пространства имён QA.Search.Generic.DAL.Services.

В контексте должны быть определены свойства с привязкой к объектам источника данных, которые планируется использовать при индексации. Стоить учесть, что GenericDataContext уже содержит ряд свойств для стандартных объектов источника данных.

Для привязки модели к объекту источника данных следует использовать вспомогательные методы из GenericDataContext:

- GetTableNameFromDotNetName – получение имени объекта в источнике данных по полю Dotnet Name источника данных;
- GetTableNameFromContentName – получение имени объекта в источнике данных по имени контента.

Пример: DemositeDataContext.

## 7.3. Создание моделей

Модели для хранения данных полученных из источника данных (БД) должны быть наследованы от класса GenericItem из пространства имён QA.Search.Generic.DAL.Models.

В классе GenericItem уже описаны так называемые «системные поля» QP8.CMS (Id, дата создания, дата изменения и прочее), в самой же модели описываются только поля специфичные для описываемого контента.

Примеры: NewsPage, TextPage и так далее.

## 7.4. Создание логики индексации данных

Для реализации логики индексации для каждого набора данных необходимо создать класс-view, наследованный от Generic-класса ElasticView<T> пространства имён QA.Search.Generic.Integration.QP.Infrastructure и в качестве Generic-а передать туда тип GenericDataContext.

Минимальный набор свойств и методов, которые необходимо переопределить для корректной работы индексатора:

- свойство IndexName;
- свойство ViewName;
- свойство Query;
- метод LoadAsync

Так же можно переопределить метод CountAsync, но в каких случаях это стоит делать будет описано далее.

### 7.4.1. IndexName

Название индекса, которое будет использоваться как часть имени индекса в Elasticsearch/OpenSearch. Должно быть уникально в рамках проекта.

Пример: NewsPage.

### 7.4.2. ViewName

Название индексатора контента, которое будет использоваться во внутренней логике приложения и log-файлах.

Пример: NewsPageView.

### 7.4.3. Query

Указание объекта контекста, который будет использоваться для индексации.

Например: ((DemositeDataContext)Db).TextPages.

Помимо указания объекта контекста для индексации можно указать дополнительные ограничения, среди которых поддерживаются следующие:

- Filter

- Pick
- Omit
- JoinOne
- JoinMany

### *Filter*

Сигнатура:

```
.Filter(Func<T, bool> predicate)
```

Позволяет фильтровать данные при выборке, например, по дате, старше определенной или по Id, больше определённого, или любые другие.

Аналог Where в Linq.

Пример:

```
.Filter(n => n.PublishDate > DateTime.Parse("2019-01-01"))
```

### *Pick*

Сигнатура:

```
.Pick(Func<T, P> selector)
```

Позволяет выбирать поля для выборки из источника данных.

Является аналогом Select в Linq.

Пример:

```
.Pick(r => new { r.Title, r.Alias })
```

### *Omit*

Сигнатура:

```
.Omit(Func<T, P> selector)
```

Позволяет исключать поля из выборки из источника данных.

Пример:

```
.Omit(n => new { n.SmallImage, n.TransliterationTitle })
```

### *JoinOne*

Сигнатура:

```
.JoinOne(Func<T, P> selector)
```

Загрузка связи OneToMany.

Пример:

```
.JoinOne(n => n.PublishSMI
        .Pick(s => s.Title))
```

### 7.4.4. JoinMany

Используется для загрузки связи OneToMany или ManyToMany.

*OneToMany*

Сигнатура:

```
.JoinMany(Func<T, ICollection<E>> selector)
```

### *ManyToMany*

Сигнатура:

```
.JoinMany(Func<T, ICollection<L>> collectionSelector, Func<L, E>
elementSelector)
```

Пример:

```
.JoinMany(n => n.Regions, l => l.MarketingRegionLinkedItem
        .Filter(r => r.DpcAlias != null)
        .Pick(r => new { r.Title, r.DpcAlias })
        .JoinOne(r => r.Parent)
            .Pick(r => new { r.Title, r.DpcAlias }));
```

#### 7.4.5. LoadAsync

Этот метод отвечает за основную логику загрузки и нормализации документов перед передачей их в Elasticsearch/OpenSearch.

Его можно поделить на две условные части:

- выгрузка данных
- нормализация данных

### *Выгрузка данных*

Выгрузку можно выполнить двумя способами.

Если функционала Query достаточно для выполнения задачи выгрузки данных из источника данных, то для получения данных достаточно вызвать метод LoadAsync базового класса. Данный метод вернёт массив JsonObject согласно запросу, который указан в Query.

Если функционала Query недостаточно, то можно вручную написать запрос к свойству контекста, используя методы Entity Framework Core и расширения к нему (Select, Where, OrderBy и т.д.). Однако в этом случае следует переопределить метод CountAsync и реализовать там схожий запрос, возвращающий кол-во записей, удовлетворяющих условиям. Так же при ручной работе с БД стоит помнить, что в результате мы получаем не массив Json-документов, а список экземпляров класса, который мы возвращаем из источника данных.

### *Нормализация данных*

После выгрузки данных и перед их отправкой в Elasticsearch/OpenSearch можно выполнять любые манипуляции над данными, исходя из требований по их сохранению.

Например, для новостей имеется заголовок и текст новости, а на странице с результатами поиска требуется выдавать краткое содержание новости. Можно выгрузить из источника данных заголовок и полный текст, создать в JSON-документе поле с кратким описанием (например, brief) и положить туда урезанную версию полного текста.

### *Формирование адреса страницы, содержащей контент*

Для корректного формирования ссылки на страницу существует вспомогательный класс UrlService, который можно подключить через DI по интерфейсу IUrlService<T>, в качестве Generic, передав ему контекст.

Интерфейс предоставляет следующие методы:

- GetUrlToPageByNameAsync – сборка Url до страницы по имени abstract item, на котором отображается индексируемый контент;
- GetUrlToPageByIdAsync – сборка Url до страницы по Id abstract item, на котором отображается индексируемый контент;
- BuildUri – собирает адрес из базовой части и query;
- UrlToJArray – помещает Url в объект JArray готовый для записи в Elasticsearch.

Адрес обязательно должен быть сформирован через этот метод, либо вручную, но в том же виде, в котором формируется внутри этого метода.

Пример формата:

```
"SearchUrl": [
  { "SearchUrl": "https://domain/path_to_resource" }
]
```

### *Индексация ролей на уровне статьи контента*

Если требуется индексировать данные с учётом ролевой модели QP8.CMS, а ограничение по ролям применяется не к контенту целиком, а к конкретным статьям контента, то необходимо через DI подключить класс PermissionService по интерфейсу IPermissionService.

Интерфейс предоставляет доступ к следующим методам:

- GetPermissions – получение ограничения ролей для статьи контента;
- AbstractItemPermissions – получение ограничения ролей для контента.

Полученный список ролей нужно положить в поле Elasticsearch/OpenSearch по которому в последующем будет осуществляться поиск через QP8.Search.API.

### 7.4.6. Регистрация индексатора

Для того, чтобы основное приложение знало обо всех реализованных функциях индексатора необходимо реализовать класс регистрации индексатора, который должен реализовывать интерфейс IServiceRegistrator с единственным методом RegisterServices.

В этом методе нужно в IServiceCollection зарегистрировать все реализованные компоненты, необходимые для работы индексатора. Как минимум:

- контекст;
- конфигурация (если она есть);
- UrlService с нужным контекстом;
- вызвать метод AddElasticViews с Generic-контекстом.

# 8. Шаблон индексации в Elasticsearch/OpenSearch

Для корректной и быстрой работы поиска с индексами ElasticSearch/OpenSearch требуется эти индексы правильным образом сконфигурировать.

Шаблон задаётся один раз перед созданием индекса и содержит в себе паттерн имени индексов, к которым он будет применяться. Например, если мы укажем паттерн «index.search.demosite.*», то все индексы с разными именами, но фиксированной частью имени «index.search.demosite.» будут создаваться на основе этого шаблона.

Это даёт гибкие возможности в настройке индексов. Можно настраивать как каждый индекс индивидуально, так и задать общую настройку индексации для всего проекта.

Пример шаблона для проекта Demosite можно увидеть в Приложении 1.

## 8.1. Словари морфологии

Для начала стоит обратить внимание на поддержку морфологии.

Если в рамках проекта требуется в поиске учитывать семантику, синонимы и прочие особенности слова как единицы языка, то требуется подключить словари hunspell в сам ElasticSearch/OpenSearch.

### 8.1.1. Подключение словаря

Словари можно найти в репозитории.

Скачиваем директории с необходимыми в проекте словарями из репозитория и помещаем их на сервере, где установлен ElasticSearch/OpenSearch в директорию «/etc/elasticsearch/hunspell/».

### 8.1.2. Добавление фильтров словарей

Для начала использования словаря требуется создать фильтр с подключенным словарём, для чего добавим в «filters» следующий блок (пример для русского языка):

```
"hunspell_russian":
{
    "locale": "ru",
    "type": "hunspell",
    "dedup": "true"
}
```

Где «locale» должно соответствовать названию директории, содержащей файлы словаря, «type» следует указать «hunspell» (для Elasticsearch/OpenSearch – это подключаемый словарь, наличие словаря будет проверено в момент создания шаблона), «dedup» указывает на необходимость удаления дубликатов из результатов работы анализатора (после того как по всему тексту индексируемого поля проанализирована семантика и построены синонимы).

#### 8.1.3. Использование фильтров словарей

Для использования словаря в анализаторах требуется указать название фильтра с нужным словарём в списке фильтров анализатора, например, следующим образом:

```
"analyzer_text":
    {
    "filter":
    [
        "lowercase",
        "stopwords_russian",
        "stopwords_english",
        "hunspell_russian",
        "hunspell_english"
    ],
    "type": "custom",
    "tokenizer": "unicode_words"
}
```

### 8.2. Основные настройки шаблона

#### 8.2.1. Паттерн разбора по регулярному выражению

Для разбора текста на отдельные блоки, по которым будет осуществляться поиск, выполняется разбор по регулярному выражению, которое настраивается в этом блоке:

```
"unicode_words":
{
    "pattern": "\\d+[\\.,]\\d+|[\\p{L}\\d_+*#]+",
    "type": "pattern",
    "group": "0"
}
```

#### 8.2.2. Анализаторы текста

Для корректной работы с текстом в шаблоне должны быть заданы анализаторы для работы с «text», «synonyms», «shingles», «prefixes» и «regex», все они указаны в секции «analyzer».

Стоит отметить, что каждый анализатор имеет свой набор зависимостей в виде фильтров, токенов и тд. Все зависимости приведены в этом же файле шаблона в соответствующих блоках.

Пример анализаторов можно посмотреть в примере шаблона из Приложение 1.

## 8.3. Настройки индексации полей

### 8.3.1. Динамические привязки

Для настройки общих привязок к ряду полей индекса в автоматическом режиме используются динамические привязки в разделе «dynamic_templates».

#### *Разбор текстовых полей*

При анализе текстовых полей динамической привязкой происходит нормализация и разбор текста на «prefixes», «synonyms», «shingles» и «phrases» с копированием их в соответствующие поля индекса.

Всё это выполняется следующей настройкой:

```
"text":
{
    "match_pattern": "regex",
    "path_match": "title|text|description",
    "mapping":
    {
        "copy_to":
        [
            "_phrases",
            "_prefixes",
            "_shingles",
            "_synonyms"
        ],
        "normalizer": "normalizer_keyword",
        "fields":
        {
            "prefixes":
            {
                "search_analyzer": "analyzer_text",
                "analyzer": "analyzer_prefixes",
                "type": "text"
            },
            "synonyms":
            {
                "search_analyzer": "analyzer_synonyms_search",
                "analyzer": "analyzer_synonyms_index",
                "type": "text"
            },
            "shingles":
            {
                "analyzer": "analyzer_shingles",
                "type": "text"
            },
            "phrases":
            {
                "analyzer": "analyzer_phrases",
                "type": "text"
            }
        },
        "type": "keyword"
    }
}
```

Здесь стоит обратить внимание на поле «path_match»: в нём задаётся список полей, которые будут попадать под анализ. Необходимо задать актуальный для проекта список полей.

### *Дополнительные динамические привязки*

Помимо привязки для разбора текстовых полей регулярным выражением описанной выше можно задавать дополнительные привязки, например, правила для разбора чисел, дат и так далее.

#### 8.3.2. Статические правила индексации

Если требуется, чтобы поле при индексации анализировалось и хранилось особым образом, то для него стоит создать правило в секции «properties».

Правила для «_phrases», «_prefixes», «_shingles», «_synonyms», «contentitemid» и «SearchUrl» должны соответствовать тем, что указаны в примере шаблона, они обеспечивают работу базовой функциональности API поиска.

Идущие далее правила можно настраивать под задачи проекта, а также можно добавлять новые правила.

### *Индексация текстовых полей*

Для наиболее эффективной индексации текстовых полей в связке с API поиска следует использовать следующий шаблон индексации:

```
"title":
{
    "copy_to":
    [
        "_phrases",
        "_prefixes",
        "_shingles",
        "_synonyms"
    ],
    "normalizer": "normalizer_keyword",
    "type": "keyword",
    "fields":
    {
        "prefixes":
        {
            "search_analyzer": "analyzer_text",
            "analyzer": "analyzer_prefixes",
            "type": "text"
        },
        "synonyms":
        {
            "search_analyzer": "analyzer_synonyms_search",
            "analyzer": "analyzer_synonyms_index",
            "type": "text"
        },
        "shingles":
        {
            "analyzer": "analyzer_shingles",
            "type": "text"
        },
        "phrases":
        {
            "analyzer": "analyzer_phrases",
            "type": "text"
        }
    }
}
```

Где «title» - название столбца для индексации, а далее стандартные правила наиболее эффективной индексации для API поиска. Правила можно дополнять и расширять, но крайне не рекомендуется урезать или убирать.

При отсутствии корректно настроенных правил индексации работоспособность поиска не гарантируется.

### *Дополнительные статически правила*

Помимо описанного выше правила индексации текстовых полей можно настраивать любые дополнительные правила для любых полей. Однако важно, чтобы описанные правила не конфликтовали с обязательными правилами и настройками.

## 8.4. Загрузка шаблона в Elasticsearch/OpenSearch

Для загрузки шаблона индекса в ElasticSearch/OpenSearch требуется выполнить «PUT» запрос в Elasticsearch/OpenSearch в Endpoint «_index_template» и через «/» указать название шаблона, например, «http://elastic:9200/_index_template/search.demosite».

Выполнять запрос можно через «curl», «Insomnia», «PostMan» или любые другие альтернативы.

Пример запроса на загрузку шаблона можно увидеть в Приложении 2.

## 8.5. Обновление шаблона

Для обновления шаблона нужно внести изменения в текст шаблона, после чего выполнить повторную отправку шаблона тем же методом, что указан в разделе «Загрузка шаблона в Elasticsearch/OpenSearch».

Стоит помнить, что уже созданные индексы не будут обновлены и данные в них не будут переиндексированы после обновления шаблона.

Для применения шаблона к индексам следует дождаться плановой полной переиндексации сервисом индексации контента, либо запустить переиндексацию вручную через веб-интерфейс администратора поиска «Search Admin App» (подробнее про запуск переиндексации можно прочитать в инструкции пользователя «Search Admin App»).

# 9. Настройка приложения

После реализации собственного модуля интеграции нужно собрать проект QA.Search.Generic.Integration.API и проект собственного модуля интеграции.

Все файлы после сборки обоих проектов положить в одну общую директорию на диске.

Заполняем файл appsettings.json.

В секции CommonSettings параметры:

- IndexPermissions – true если требуется индексация ролей QP8.CMS для контентов;
- IndexerLibraries – массив библиотек индексации в формате «имя файла» без расширения (например, «QA.Search.Integration.Demosite»).
- в секции ElasticSettings:
- Address – адрес и порт по которому доступен Elasticsearch/OpenSearch;
- ProjectName – имя проекта (например Demosite), должен совпадать с такой настройках у остальных компонентов QP8.Search;
- RequestTimeout – время ожидания ответа от Elasticsearch/OpenSearch.

В секциях Settings.QP, Settings.QP.Update и Settings.QP.Permissions:

- CronSchedule – расписание в формате cron с какой частотой запускать каждую индексацию.

В секции PermissionsConfiguration:

- DefaultRoleAlias – имя роли для контентов и статей доступных всем;
- PermissionIndexName – часть названия индекса, куда будут сохраняться индексы и роли доступа в них;
- QPAbstractItems – названия Abstract item-ов QP8.CMS для которых требуется индексировать разрешения по ролям (подробнее узнать про QPAbstractItems можно в документации пользователя по продукту QP8.CMS).

В секции ViewOptions подсекции ViewParameters для каждой созданной View создать объект с именем View и параметром BatchSize, в котором указать числом кол-во данных, которые требуется выбирать из источника данных за один раз.

В секции ContextConfiguration:

- ConnectionString – строка подключения к БД;
- SqlServerType – MsSql или PostgreSQL;
- DefaultSchemeName – название схемы, используемой в БД (например, public);
- ContentAccess – тип контента, который будет использоваться для выборки информации (Stage или Live, подробнее про разделение на Stage и Live можно узнать в документации пользователя по продукту QP8.CMS);
- FormatTableName – формат именования таблиц (например, «{0}.{1}» для формата schema.table).

# Приложение 1

Пример шаблона Elasticsearch/OpenSearch, реализованного для референсной интеграции QP8.Search с проектом Demosite.

```
{
  "index_patterns": [
    "index.search.demosite.*"
  ],
  "template": {
    "settings": {
      "index": {
        "mapping": {
          "total_fields": {
            "limit": "100000"
          }
        },
        "analysis": {
          "filter": {
            "stemmer_english": {
              "type": "stemmer",
              "language": "english"
            },
            "synonyms_index": {
              "type": "synonym_graph",
              "synonyms": []
            },
            "stopwords_russian": {
              "type": "stop",
              "stopwords": "_russian_"
            },
            "big_shingles": {
              "max_shingle_size": "4",
              "min_shingle_size": "2",
              "type": "shingle"
            },
            "shingles": {
              "max_shingle_size": "3",
              "min_shingle_size": "2",
              "type": "shingle"
            },
            "stopwords_english": {
              "type": "stop",
              "stopwords": "_english_"
            },
            "synonyms_search": {
              "type": "synonym_graph",
```

```
            "synonyms": []
          },
          "stemmer_russian": {
            "type": "stemmer",
            "language": "russian"
          },
          "hunspell_russian": {
            "locale": "ru",
            "type": "hunspell",
            "dedup": "true"
          },
          "edge_ngram": {
            "type": "edge_ngram",
            "min_gram": "2",
            "max_gram": "20"
          },
          "hunspell_english": {
            "locale": "en_GB",
            "type": "hunspell",
            "dedup": "true"
          }
        },
        "char_filter": {
          "collapse_white_space": {
            "pattern": "[\\p{Z}\\r\\n]+",
            "type": "pattern_replace",
            "replacement": " "
          },
          "truncate_keyword": {
            "pattern": "^\\p{Z}*(.{2045}).*$",
            "type": "pattern_replace",
            "replacement": "$1..."
          }
        },
        "normalizer": {
          "normalizer_keyword": {
            "type": "custom",
            "char_filter": [
              "collapse_white_space",
              "truncate_keyword"
            ]
          }
        },
        "analyzer": {
          "analyzer_shingles": {
            "filter": [
              "lowercase",
```

```
            "stopwords_russian",
            "stopwords_english",
            "hunspell_russian",
            "hunspell_english",
            "shingles"
          ],
          "char_filter": [
            "html_strip"
          ],
          "type": "custom",
          "tokenizer": "unicode_words"
        },
        "analyzer_text": {
          "filter": [
            "lowercase",
            "stopwords_russian",
            "stopwords_english",
            "hunspell_russian",
            "hunspell_english"
          ],
          "type": "custom",
          "tokenizer": "unicode_words"
        },
        "analyzer_prefixes": {
          "filter": [
            "lowercase",
            "stopwords_russian",
            "stopwords_english",
            "hunspell_russian",
            "hunspell_english",
            "edge_ngram"
          ],
          "char_filter": [
            "html_strip"
          ],
          "type": "custom",
          "tokenizer": "unicode_words"
        },
        "analyzer_regex": {
          "filter": [
            "lowercase",
            "stopwords_english",
            "stopwords_russian",
            "stemmer_english",
            "stemmer_russian"
          ],
          "type": "custom",
```

```
            "tokenizer": "unicode_words"
          },
          "analyzer_phrases": {
            "filter": [
              "lowercase",
              "big_shingles"
            ],
            "char_filter": [
              "html_strip"
            ],
            "type": "custom",
            "tokenizer": "unicode_words"
          },
          "analyzer_synonyms_index": {
            "filter": [
              "lowercase",
              "stopwords_russian",
              "stopwords_english",
              "hunspell_russian",
              "hunspell_english",
              "synonyms_index"
            ],
            "char_filter": [
              "html_strip"
            ],
            "type": "custom",
            "tokenizer": "unicode_words"
          },
          "analyzer_synonyms_search": {
            "filter": [
              "lowercase",
              "stopwords_russian",
              "stopwords_english",
              "hunspell_russian",
              "hunspell_english",
              "synonyms_search"
            ],
            "type": "custom",
            "tokenizer": "unicode_words"
          }
        },
        "tokenizer": {
          "unicode_words": {
            "pattern": "\\d+[\\.,]\\d+|[\\p{L}\\d_+*#]+",
            "type": "pattern",
            "group": "0"
          }
```

```
        }
      },
      "number_of_shards": "1",
      "number_of_replicas": "2"
    }
  },
  "mappings": {
    "dynamic_date_formats": [
      "MM/dd/yyyy",
      "MM/dd/yyyy HH:mm",
      "MM/dd/yyyy HH:mm:ss",
      "dd.MM.yyyy",
      "dd.MM.yyyy HH:mm",
      "dd.MM.yyyy HH:mm:ss",
      "yyyy-MM-dd",
      "yyyy-MM-dd'T'HH:mm",
      "yyyy-MM-dd'T'HH:mm:ss",
      "yyyy-MM-dd'T'HH:mm:ssZZZZZ",
      "yyyy-MM-dd'T'HH:mm:ss.SSS",
      "yyyy-MM-dd'T'HH:mm:ss.SSSZZZZZ"
    ],
    "dynamic_templates": [
      {
        "text": {
          "match_pattern": "regex",
          "path_match": "title|text|description",
          "mapping": {
            "copy_to": [
              "_phrases",
              "_prefixes",
              "_shingles",
              "_synonyms"
            ],
            "normalizer": "normalizer_keyword",
            "fields": {
              "prefixes": {
                "search_analyzer": "analyzer_text",
                "analyzer": "analyzer_prefixes",
                "type": "text"
              },
              "synonyms": {
                "search_analyzer": "analyzer_synonyms_search",
                "analyzer": "analyzer_synonyms_index",
                "type": "text"
              },
              "shingles": {
                "analyzer": "analyzer_shingles",
```

```
                "type": "text"
              },
              "phrases": {
                "analyzer": "analyzer_phrases",
                "type": "text"
              }
            },
            "type": "keyword"
          }
        }
      },
      {
        "nested": {
          "match_pattern": "regex",
          "mapping": {
            "type": "nested"
          },
          "match_mapping_type": "object",
          "match": "Children"
        }
      },
      {
        "number": {
          "match_mapping_type": "long",
          "mapping": {
            "type": "double"
          }
        }
      },
      {
        "string": {
          "match_mapping_type": "string",
          "mapping": {
            "type": "keyword",
            "normalizer": "normalizer_keyword"
          }
        }
      }
    ],
    "properties": {
      "_phrases": {
        "type": "text",
        "analyzer": "analyzer_phrases"
      },
      "_prefixes": {
        "type": "text",
        "analyzer": "analyzer_prefixes",
```

```
        "search_analyzer": "analyzer_text",
        "store": true
      },
      "_shingles": {
        "type": "text",
        "analyzer": "analyzer_shingles",
        "store": true
      },
      "_synonyms": {
        "type": "text",
        "analyzer": "analyzer_synonyms_index",
        "search_analyzer": "analyzer_synonyms_search"
      },
      "contentitemid": {
        "type": "double"
      },
      "SearchUrl": {
        "properties": {
          "SearchUrl": {
            "normalizer": "normalizer_keyword",
            "type": "keyword"
          }
        }
      },
      "description": {
        "copy_to": [
          "_phrases",
          "_prefixes",
          "_shingles",
          "_synonyms"
        ],
        "normalizer": "normalizer_keyword",
        "type": "keyword",
        "fields": {
          "prefixes": {
            "search_analyzer": "analyzer_text",
            "analyzer": "analyzer_prefixes",
            "type": "text"
          },
          "synonyms": {
            "search_analyzer": "analyzer_synonyms_search",
            "analyzer": "analyzer_synonyms_index",
            "type": "text"
          },
          "shingles": {
            "analyzer": "analyzer_shingles",
            "type": "text"
```

```
        },
        "phrases": {
          "analyzer": "analyzer_phrases",
          "type": "text"
        }
      }
    },
    "title": {
      "copy_to": [
        "_phrases",
        "_prefixes",
        "_shingles",
        "_synonyms"
      ],
      "normalizer": "normalizer_keyword",
      "type": "keyword",
      "fields": {
        "prefixes": {
          "search_analyzer": "analyzer_text",
          "analyzer": "analyzer_prefixes",
          "type": "text"
        },
        "synonyms": {
          "search_analyzer": "analyzer_synonyms_search",
          "analyzer": "analyzer_synonyms_index",
          "type": "text"
        },
        "shingles": {
          "analyzer": "analyzer_shingles",
          "type": "text"
        },
        "phrases": {
          "analyzer": "analyzer_phrases",
          "type": "text"
        }
      }
    },
    "text": {
      "copy_to": [
        "_phrases",
        "_prefixes",
        "_shingles",
        "_synonyms"
      ],
      "normalizer": "normalizer_keyword",
      "type": "keyword",
      "fields": {
```

```
            "prefixes": {
              "search_analyzer": "analyzer_text",
              "analyzer": "analyzer_prefixes",
              "type": "text"
            },
            "synonyms": {
              "search_analyzer": "analyzer_synonyms_search",
              "analyzer": "analyzer_synonyms_index",
              "type": "text"
            },
            "shingles": {
              "analyzer": "analyzer_shingles",
              "type": "text"
            },
            "phrases": {
              "analyzer": "analyzer_phrases",
              "type": "text"
            }
          }
        },
        "description": {
          "copy_to": [
            "_phrases",
            "_prefixes",
            "_shingles",
            "_synonyms"
          ],
          "normalizer": "normalizer_keyword",
          "type": "keyword",
          "fields": {
            "prefixes": {
              "search_analyzer": "analyzer_text",
              "analyzer": "analyzer_prefixes",
              "type": "text"
            },
            "synonyms": {
              "search_analyzer": "analyzer_synonyms_search",
              "analyzer": "analyzer_synonyms_index",
              "type": "text"
            },
            "shingles": {
              "analyzer": "analyzer_shingles",
              "type": "text"
            },
            "phrases": {
              "analyzer": "analyzer_phrases",
              "type": "text"
```

```
                }
            }
        }
    }
  }
}
```

## Приложение 2

Пример запроса на загрузку шаблона в Elasticsearch/OpenSearch через программу «curl».

```
curl --request PUT \
  --url http://127.0.0.1:9200/_index_template/search.demosite \
  --header 'Content-Type: application/json' \
  --data '{
      "index_patterns": [
            "index.search.demosite.*"
      ],
      "template": {
            "settings": {
                  "index": {
                        "mapping": {
                              "total_fields": {
                                    "limit": "100000"
                              }
                        },
                        "analysis": {
                              "filter": {
                                    "stemmer_english": {
                                          "type": "stemmer",
                                          "language": "english"
                                    },
                                    "synonyms_index": {
                                          "type": "synonym_graph",
                                          "synonyms": []
                                    },
                                    "stopwords_russian": {
                                          "type": "stop",
                                          "stopwords": "_russian_"
                                    },
                                    "big_shingles": {
                                          "max_shingle_size": "4",
                                          "min_shingle_size": "2",
                                          "type": "shingle"
                                    },
                                    "shingles": {
                                          "max_shingle_size": "3",
                                          "min_shingle_size": "2",
                                          "type": "shingle"
                                    },
                                    "stopwords_english": {
                                          "type": "stop",
                                          "stopwords": "_english_"
```

```
                        },
                        "synonyms_search": {
                              "type": "synonym_graph",
                              "synonyms": []
                        },
                        "stemmer_russian": {
                              "type": "stemmer",
                              "language": "russian"
                        },
                        "hunspell_russian": {
                              "locale": "ru",
                              "type": "hunspell",
                              "dedup": "true"
                        },
                        "edge_ngram": {
                              "type": "edge_ngram",
                              "min_gram": "2",
                              "max_gram": "20"
                        },
                        "hunspell_english": {
                              "locale": "en_GB",
                              "type": "hunspell",
                              "dedup": "true"
                        }
                  },
                  "char_filter": {
                        "collapse_white_space": {
                              "pattern": "[\\p{Z}\\r\\n]+",
                              "type": "pattern_replace",
                              "replacement": " "
                        },
                        "truncate_keyword": {
                              "pattern": "^\\p{Z}*(.{2045}).*$",
                              "type": "pattern_replace",
                              "replacement": "$1..."
                        }
                  },
                  "normalizer": {
                        "normalizer_keyword": {
                              "type": "custom",
                              "char_filter": [
                                    "collapse_white_space",
                                    "truncate_keyword"
                              ]
                        }
                  },
                  "analyzer": {
```

```
                        "analyzer_shingles": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_russian",
                                    "stopwords_english",
                                    "hunspell_russian",
                                    "hunspell_english",
                                    "shingles"
                              ],
                              "char_filter": [
                                    "html_strip"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_text": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_russian",
                                    "stopwords_english",
                                    "hunspell_russian",
                                    "hunspell_english"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_prefixes": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_russian",
                                    "stopwords_english",
                                    "hunspell_russian",
                                    "hunspell_english",
                                    "edge_ngram"
                              ],
                              "char_filter": [
                                    "html_strip"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_regex": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_english",
                                    "stopwords_russian",
                                    "stemmer_english",
```

```
                                    "stemmer_russian"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_phrases": {
                              "filter": [
                                    "lowercase",
                                    "big_shingles"
                              ],
                              "char_filter": [
                                    "html_strip"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_synonyms_index": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_russian",
                                    "stopwords_english",
                                    "hunspell_russian",
                                    "hunspell_english",
                                    "synonyms_index"
                              ],
                              "char_filter": [
                                    "html_strip"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        },
                        "analyzer_synonyms_search": {
                              "filter": [
                                    "lowercase",
                                    "stopwords_russian",
                                    "stopwords_english",
                                    "hunspell_russian",
                                    "hunspell_english",
                                    "synonyms_search"
                              ],
                              "type": "custom",
                              "tokenizer": "unicode_words"
                        }
                  },
                  "tokenizer": {
                        "unicode_words": {
```

```
                                        "pattern":
"\\d+[\\.,]\\d+|[\\p{L}\\d_+*#]+",
                                        "type": "pattern",
                                        "group": "0"
                                  }
                            }
                      },
                      "number_of_shards": "1",
                      "number_of_replicas": "2"
                }
          },
          "mappings": {
                "dynamic_date_formats": [
                      "MM/dd/yyyy",
                      "MM/dd/yyyy HH:mm",
                      "MM/dd/yyyy HH:mm:ss",
                      "dd.MM.yyyy",
                      "dd.MM.yyyy HH:mm",
                      "dd.MM.yyyy HH:mm:ss",
                      "yyyy-MM-dd",
                      "yyyy-MM-dd'\''T'\''HH:mm",
                      "yyyy-MM-dd'\''T'\''HH:mm:ss",
                      "yyyy-MM-dd'\''T'\''HH:mm:ssZ",
                      "yyyy-MM-dd'\''T'\''HH:mm:ss.SSS",
                      "yyyy-MM-dd'\''T'\''HH:mm:ss.SSSZ"
                ],
                "dynamic_templates": [
                      {
                            "text": {
                                  "match_pattern": "regex",
                                  "path_match": "title|text|description",
                                  "mapping": {
                                        "copy_to": [
                                              "_phrases",
                                              "_prefixes",
                                              "_shingles",
                                              "_synonyms"
                                        ],
                                        "normalizer": "normalizer_keyword",
                                        "fields": {
                                              "prefixes": {
                                                    "search_analyzer":
"analyzer_text",
                                                    "analyzer":
"analyzer_prefixes",
                                                    "type": "text"
                                              },
```

```
                                                "synonyms": {
                                                      "search_analyzer":
"analyzer_synonyms_search",
                                                      "analyzer":
"analyzer_synonyms_index",
                                                      "type": "text"
                                                },
                                                "shingles": {
                                                      "analyzer":
"analyzer_shingles",
                                                      "type": "text"
                                                },
                                                "phrases": {
                                                      "analyzer":
"analyzer_phrases",
                                                      "type": "text"
                                                }
                                          },
                                          "type": "keyword"
                                    }
                              }
                        },
                        {
                              "nested": {
                                    "match_pattern": "regex",
                                    "mapping": {
                                          "type": "nested"
                                    },
                                    "match_mapping_type": "object",
                                    "match": "Children"
                              }
                        },
                        {
                              "number": {
                                    "match_mapping_type": "long",
                                    "mapping": {
                                          "type": "double"
                                    }
                              }
                        },
                        {
                              "string": {
                                    "match_mapping_type": "string",
                                    "mapping": {
                                          "type": "keyword",
                                          "normalizer": "normalizer_keyword"
                                    }
```

```
                        }
                    }
                ],
                "properties": {
                    "_phrases": {
                        "type": "text",
                        "analyzer": "analyzer_phrases"
                    },
                    "_prefixes": {
                        "type": "text",
                        "analyzer": "analyzer_prefixes",
                        "search_analyzer": "analyzer_text",
                        "store": true
                    },
                    "_shingles": {
                        "type": "text",
                        "analyzer": "analyzer_shingles",
                        "store": true
                    },
                    "_synonyms": {
                        "type": "text",
                        "analyzer": "analyzer_synonyms_index",
                        "search_analyzer": "analyzer_synonyms_search"
                    },
                    "contentitemid": {
                        "type": "double"
                    },
                    "SearchUrl": {
                        "properties": {
                            "SearchUrl": {
                                "normalizer": "normalizer_keyword",
                                "type": "keyword"
                            }
                        }
                    },
                    "title": {
                        "copy_to": [
                            "_phrases",
                            "_prefixes",
                            "_shingles",
                            "_synonyms"
                        ],
                        "normalizer": "normalizer_keyword",
                        "type": "keyword",
                        "fields": {
                            "prefixes": {
                                "search_analyzer": "analyzer_text",
```

```
                                    "analyzer": "analyzer_prefixes",
                                    "type": "text"
                              },
                              "synonyms": {
                                    "search_analyzer":
"analyzer_synonyms_search",
                                    "analyzer":
"analyzer_synonyms_index",
                                    "type": "text"
                              },
                              "shingles": {
                                    "analyzer": "analyzer_shingles",
                                    "type": "text"
                              },
                              "phrases": {
                                    "analyzer": "analyzer_phrases",
                                    "type": "text"
                              }
                        }
                  },
                  "text": {
                        "copy_to": [
                              "_phrases",
                              "_prefixes",
                              "_shingles",
                              "_synonyms"
                        ],
                        "normalizer": "normalizer_keyword",
                        "type": "keyword",
                        "fields": {
                              "prefixes": {
                                    "search_analyzer": "analyzer_text",
                                    "analyzer": "analyzer_prefixes",
                                    "type": "text"
                              },
                              "synonyms": {
                                    "search_analyzer":
"analyzer_synonyms_search",
                                    "analyzer":
"analyzer_synonyms_index",
                                    "type": "text"
                              },
                              "shingles": {
                                    "analyzer": "analyzer_shingles",
                                    "type": "text"
                              },
                              "phrases": {
```

```
                                        "analyzer": "analyzer_phrases",
                                        "type": "text"
                                  }
                            }
                      },
                      "description": {
                            "copy_to": [
                                  "_phrases",
                                  "_prefixes",
                                  "_shingles",
                                  "_synonyms"
                            ],
                            "normalizer": "normalizer_keyword",
                            "type": "keyword",
                            "fields": {
                                  "prefixes": {
                                        "search_analyzer": "analyzer_text",
                                        "analyzer": "analyzer_prefixes",
                                        "type": "text"
                                  },
                                  "synonyms": {
                                        "search_analyzer":
"analyzer_synonyms_search",
                                        "analyzer":
"analyzer_synonyms_index",
                                        "type": "text"
                                  },
                                  "shingles": {
                                        "analyzer": "analyzer_shingles",
                                        "type": "text"
                                  },
                                  "phrases": {
                                        "analyzer": "analyzer_phrases",
                                        "type": "text"
                                  }
                            }
                      }
                }
          }
     }
}'
```